TURING 图灵程序设计丛书

MANNING

算法图解

像小说一样有趣的算法入门书

[美] Aditya Bhargava ◎著 袁国忠 ◎译



中国工信出版集团



人民邮电出版社
POSTS & TELECOM PRESS

[illegible]

[illegible]

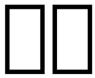
[illegible][[illegible]] Aditya Bhargava

[illegible]

ISBN[illegible]978-7-115-44763-0

[illegible]

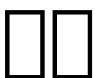
[illegible]Sangeeta[illegible]Yogesh



Visual Basic 6 for Dummies

Git “ ”

Manning Manning



Manning Marjan Bace Mike Stephens Bert Bates Jennifer Stout Manning Kevin Sullivan Mary Piergies Tiffany Taylor Leslie Haimes Karen Bensdon Rob Green Michael Hamrah Ozren Harlovic Colin Hastie Christopher Haupt Chuck Henderson Pawel Kozlowski Amit Lamba Jean-Francois Morin Robert Morrison Sankar Ramanathan Sander Rossel Doug Sparling Damien White

[illegible]Flaskhit[illegible]Ben Vinegar[illegible]Karl Puzon[illegible]Alex Manning[illegible]Esther Chan[illegible]Anish Bhatt[illegible]Michael Glass[illegible]Nikrad Mahdi[illegible]Charles Lee[illegible]Jared Friedman[illegible]Hema Manickavasagam[illegible]Hari Raja[illegible]Murali Gudipati[illegible]Srinivas Varadan[illegible]Gerry Brady[illegible]CLRS[1] [illegible]Strang[illegible]

> [1] [illegible]Thomas H. Cormen[illegible]Charles E. Leiserson[illegible]Ronald L. Rivest[illegible]Clifford Stein[illegible]——[illegible]

[illegible]Priyanka[illegible]Maggie[illegible]

[illegible]

[illegible]

[illegible]

[illegible]2[illegible]

[illegible]

[illegible]

[illegible]

- [illegible]1[illegible]——[illegible]O[illegible]O[illegible]

- [illegible]2[illegible]——[illegible]5[illegible]

- [illegible]3[illegible]4[illegible]

[illegible]O[illegible]

[illegible]

- [illegible]4[illegible]8[illegible]9[illegible]4[illegible]9[illegible]8[illegible]

- [illegible]5[illegible]

- [illegible]6[illegible]7[illegible]6[illegible]7[illegible]

- K[illegible]KNN[illegible]10[illegible]OCR[illegible]“[illegible]Adit[illegible]4[illegible]”[illegible]“[illegible]Q”[illegible]

- [illegible]11[illegible]10[illegible]

[illegible]

[illegible]

[illegible]
www.manning.com/books/grokking-algorithms [illegible]
https://github.com/egonschiele/grokking_algorithms [illegible]

[illegible]5~10[illegible]

[illegible]

[illegible]

- [illegible]
- [illegible]
- [illegible]
- [illegible]

[illegible]

[illegible]Python 2.7[illegible]

[illegible]
https://github.com/egonschiele/grokking_algorithms [illegible]

[illegible]

[illegible]

[illegible]Manning[illegible]

www.manning.com/books/grokking-algorithms [illegible]

Manning[illegible]Manning[illegible]

[illegible] 1 [illegible]

[illegible]

- [illegible]
- [illegible]——[illegible]
- [illegible]——[illegible]O[illegible]
- [illegible]——[illegible]

1.1 [illegible]

[illegible]

- [illegible]1[illegible]40 [illegible]32[illegible]
- GPS[illegible]6[illegible]7[illegible]8[illegible]
- [illegible]AI[illegible]9[illegible]

[illegible]O[illegible]

1.1.1 □□□□

□□□
□□
□□□

1.1.2 □□□□□□

□□□□□□□□□□□□□□□□□□□□

- □□□□□□□□□□□□□□□□□□□□□□AI□□□
- □□□□□K□□□□□□□□□□□
- □□□□□□□□□□□□□□□□□□□NP□□□□□□□□□□□□□□□□□□□□□□
  □□□□□□□□□□□□

□□□AI□□□
□□□□□□□□□□□□□□□□□□□□

□□□□□□

□□□□□□□□□□□□□□□□□□□□□□□□$f(x) = x \times 2$□$f(5)$□□□□
□□□□□□□□10□□□□□

□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□Python□
□□□□□□□□□□□□□□□□□□□□□Python□□□□□□□□□□□□□□□□
□□□Ruby□□□□□□□□□□□

1.2 □□□□

□□□□□□□□□□□□□K□□□□□□□□□□□□□□□□□□□□□□□□□□□K□□□□
□□□□□□□□□□□□□□□□□□□□□□□K□□□□□□□□□□□

[illegible]O[illegible]

[illegible]Facebook[illegible]Facebook[illegible]karlmageddon[illegible]Facebook[illegible]A[illegible]

[illegible]

[illegible]null[illegible]

[illegible]

"KRAMERICA 实业公司"

电话簿中有 KRAMERICA 实业公司，第 1440 条就是

1440

"VANDELAY 实业公司"

NULL

使用二分查找在电话簿中查找公司

[illegible]1[illegible]100[illegible]

1	2	3	....	100

[illegible]

[illegible]1[illegible]

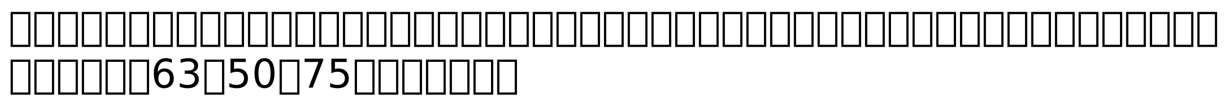
一种糟糕的猜数法

[illegible] [illegible] [illegible]99[illegible]99[illegible]

1.2.1 [illegible]

[illegible]50[illegible]

[illegible] [illegible]1[illegible]50[illegible]75[illegible]







[illegible]7[illegible]

[illegible]240 000[illegible]

简单查找：_____步

二分查找：_____步

□□□□□□□□□□□□□□□□□□□□□□□□240 000□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□

□□□□□□□□□□□18□——□□□□□□□□□□□□□n□□□□□□□□□□□□□□□□$\log_2 n$□□□□□□□□□□□n□□

□□

□□□□□□□□□□□□□□□□□□□□□□□□$\log_{10} 100$□□□“□□□10□□□□□100”□□□□□□□$10 \times 10 = 100$□□□□$\log_{10} 100 = 2$□□□□□□□□□□□□□□

$$10^2 = 100 \leftrightarrow \log_{10} 100 = 2$$

$$10^3 = 1000 \leftrightarrow \log_{10} 1000 = 3$$

$$2^3 = 8 \leftrightarrow \log_2 8 = 3$$

$$2^4 = 16 \leftrightarrow \log_2 16 = 4$$

$$2^5 = 32 \leftrightarrow \log_2 32 = 5$$

[illegible]

[illegible]O[illegible]log[illegible]log_2 [illegible]
[illegible]8[illegible]8[illegible]
[illegible]log *n* [illegible]8[illegible]3[illegible]log 8
= 3[illegible]2^3 = 8[illegible]1024[illegible]10[illegible]log
1024 = 10[illegible]2^{10} =1024[illegible]

[illegible]

[illegible]log[illegible]
[illegible]khanacademy.org[illegible]

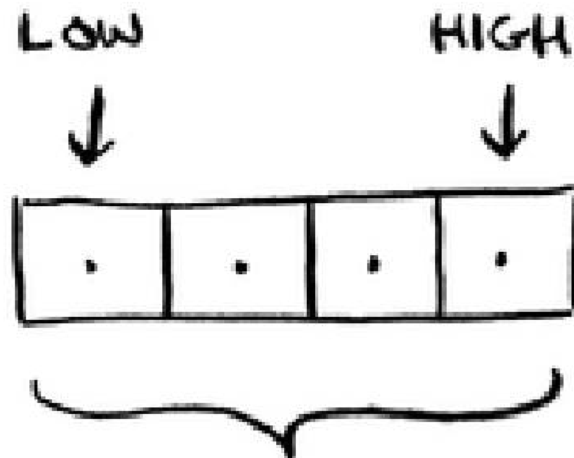
[illegible]

[illegible]
[illegible]

[illegible]Python[illegible]
[illegible]bucket[illegible]
0[illegible]#0[illegible]#1[illegible]#2[illegible]

[illegible]`binary_search` [illegible]
[illegible]——[illegible]

```
low = 0
high = len(list) - 1
```

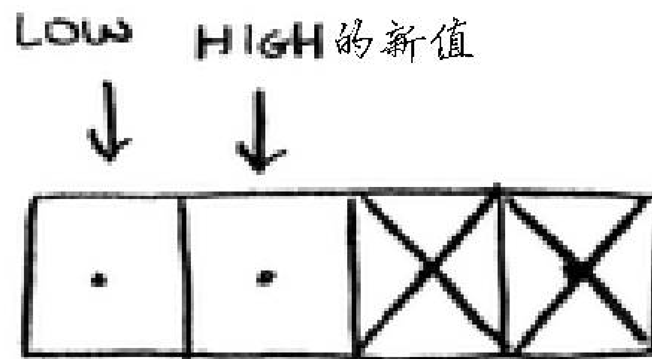


[illegible]

```
mid = (low + high) / 2  ←---[illegible](low + high)[illegible]Python[illegible]mid[illegible]
guess = list[mid]
```

[illegible]low[illegible]

```
if guess < item:
  low = mid + 1
```



[illegible]high[illegible]

```
def binary_search(list, item):
  low = 0    □□□2□□low□high□□□□□□□□□□□□□□□
  high = len(list)—1

  while low <= high:  ←-------------□□□□□□□□□□□□□□□□□
    mid = (low + high) / 2  ←-------------□□□□□□□□
    guess = list[mid]
    if guess == item:  ←-------------□□□□□
      return mid
    if guess > item:  ←-------------□□□□□□
      high = mid - 1
    else:  ←---------------------------□□□□□□
      low = mid + 1
  return None  ←--------------------□□□□□□□

my_list = [1, 3, 5, 7, 9]  ←------------□□□□□□

print binary_search(my_list, 3) # => 1  ←--------------------□□□□□□
0□□□□□□□□□□□□1
print binary_search(my_list, -1) # => None  ←--------------------□
Python□□None□□□□□□□□□□□□□□□□□
```

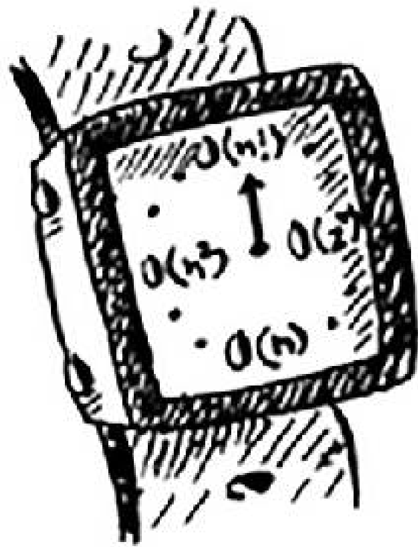
□□

1.1 □□□□□□□128□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□ □□□□□□□□
□□□□□

1.2 □□□□□□□□□□□□□□□□□□□□

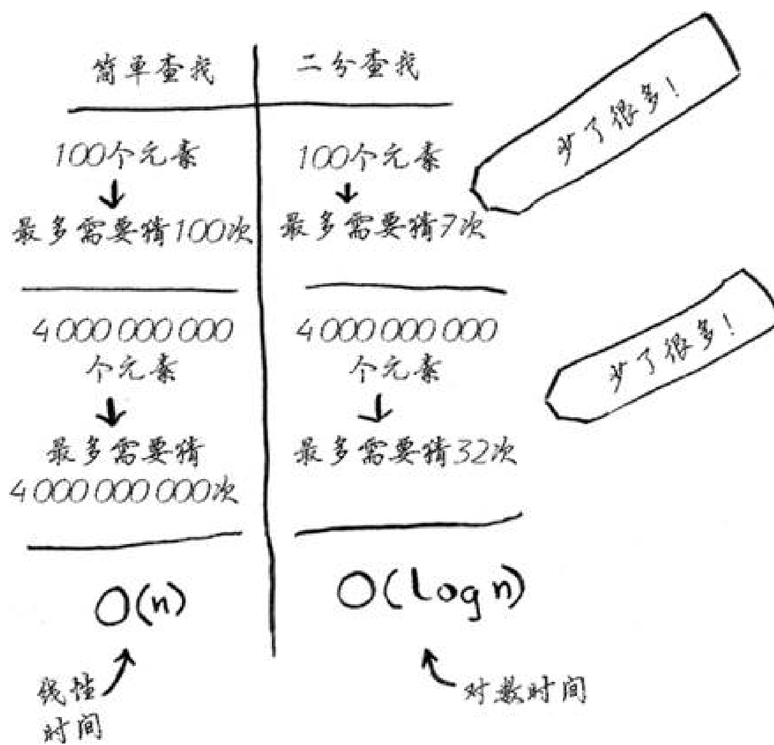
1.2.2 □□□□

□□□
□□



□□100□□□□□□□□□100□□□□□□□□40□□□□□□□□□□40□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□ □linear time□□

□□□□□□□□□□□□□□100□□□□□□□□7□□□□□□□□40□□□□□□□□□32□□□□□□□□□□□□□□□□□□□ □□log□□□□□□□□□□□□□□□□□



查找算法的运行时间

1.3 □O□□□

□O□□□ □□
□□□□□□□□□□□□□□□□□□□□□□□□□□□□□O□□□□□□□□□□□□□□□□□□□□□□□□□
□□□

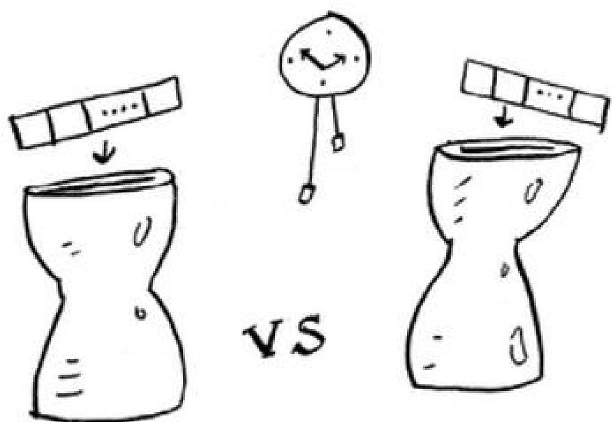
1.3.1 □□□□□□□□□□□□□□□

Bob□□NASA□□



□□□□□□□□□□□□□□□□□□□□□□□□□□Bob□□□□□□□□□□□□□□□□□□□□□□□□□□
□□□□□□□□□□□□□□□□□□□□□□□□□□Bob□□10□□□□□□□□□□□□□□□□□□□□
□□□□□□□□□□□□□□□□□□□□□□□□□□□bug□□□□□□□Bob□□□□□□□□□□□□□
□□□bug□□□□□□□□□□Bob□□□□□□□□□□□□□□100□□□□□□□□□□□□□□

[illegible]1[illegible]Bob[illegible]100[illegible]100[illegible]7[illegible]$\log_2 100$[illegible]7[illegible]7[illegible]10[illegible]



列表包含100个元素时，简单查找和二分查找的运行时间

Bob[illegible]10[illegible]30[illegible]$\log_2 1\,000\,000\,000$[illegible]30[illegible]15[illegible]100[illegible]100[illegible]7[illegible]10[illegible]$30 \times 15 = 450$[illegible]10[illegible]Bob[illegible]

[illegible]Bob[illegible]10[illegible]10[illegible]11[illegible]

	简单查找	二分查找
100 个元素	100毫秒	7毫秒
10 000 个元素	10 秒	14 毫秒
1 000 000 000 个元素	11 天	30 毫秒

运行时间的增速有天壤之别！

[illegible]Bob[illegible]15[illegible]

10[illegible]3300[illegible]
[illegible]O[illegible]

[illegible]O[illegible]n[illegible]n[illegible]O[illegible]$O(n)$[illegible]——[illegible]O[illegible]**O**[illegible]

[illegible]n[illegible]$\log n$[illegible]O[illegible]$O(\log n)$[illegible]O[illegible]

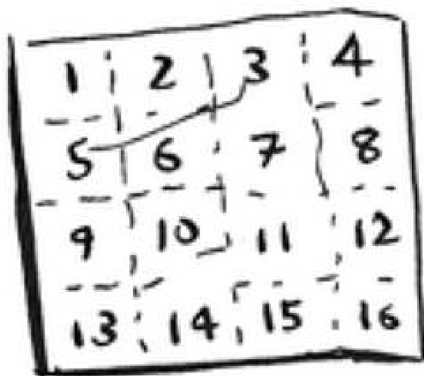
大O表示法是什么样子

[illegible]O[illegible]O[illegible]

[illegible]

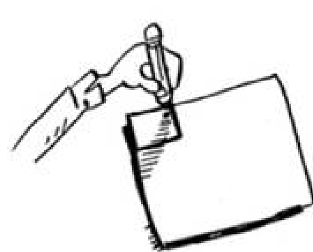
1.3.2 [illegible]O[illegible]

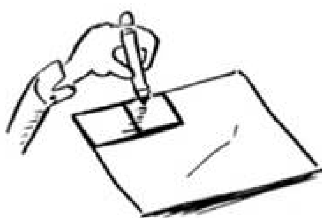
[illegible]16[illegible]

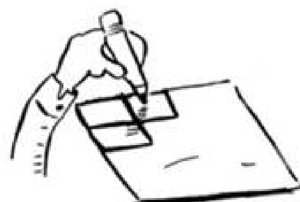


[illegible]1

[illegible]16[illegible]O[illegible]16[illegible]







[illegible]16[illegible]16[illegible]

[illegible]2

[illegible]——[illegible]

[illegible]

[illegible]4[illegible]4[illegible]16[illegible]

[illegible]4[illegible]“[illegible]”[illegible]16[illegible]

[illegible]1[illegible]$O(n)$[illegible]2[illegible]$O(\log n)$[illegible]

1.3.3 [illegible]O[illegible]

[illegible]$O(n)$[illegible]Adit——[illegible]Adit[illegible]$O(n)$[illegible]$O(1)$[illegible]

[illegible]$O(n)$[illegible]Adit[illegible]O[illegible]$O(n)$[illegible]——[illegible]$O(n)$[illegible]

[illegible]

[illegible]4[illegible]

1.3.4 [illegible]O[illegible]

[illegible]5[illegible]O[illegible]

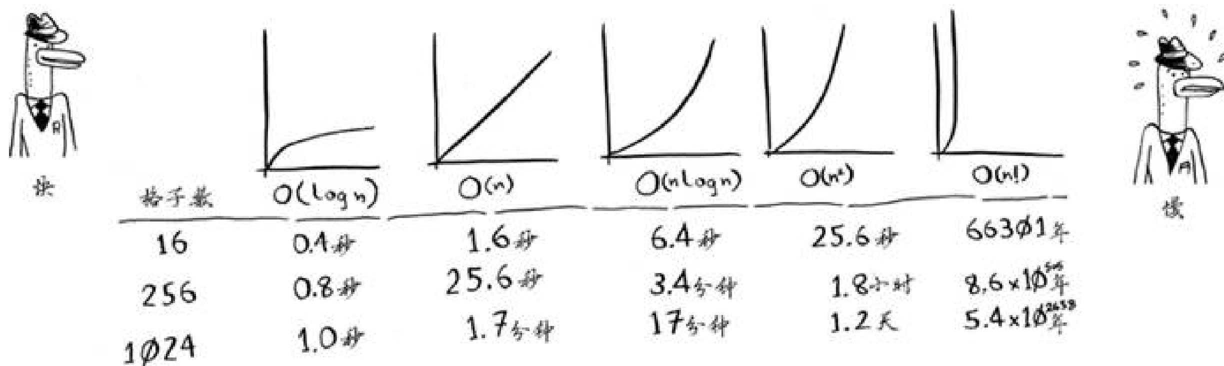
- $O(\log n)$[illegible]
- $O(n)$[illegible]
- $O(n * \log n)$[illegible]4[illegible]——[illegible]

- $O(n^2)$□□□□□□□□□2□□□□□□□□□——□□□□□□□□□□□□
- $O(n!)$□□□□□□□□□□□□□□□□□□□□□□□□□——□□□□□□□□□

□□□□□□□□□□16□□□□□□□5□□4□log 16 = 4□□□□□□□□□□□10□□□□□□□□□□□□0.4□□□□□□□□□□□1024□□□□□□□□□□□10□log 1024 = 10□□□□□□□□□□□□□□□□□□1□□□□□□□□□□□□□□□

□□□□□□□□□□□□□□$O(n)$□□□□□16□□□□□□□□16□□□□□□□1024□□□□□□□□1024□□□□□□□□□□□□□□□□□

□□□□□□□□□□□□□□□□□□□□□□□□□□□□□



□□□□□□□□□□□□5□□□□□□□

□□□□□□□□□□□□□□□□□□□□□□□O□□4□□□□□□□□□□□O□□□□□□□□□□□□□□□□□□□

- □□□□□□□□□□□□□□□□□□□□□
- □□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□
- □□□□□□□□□O□□□□□□
- $O(\log n)$□$O(n)$□□□□□□□□□□□□□□□□□□□□□□□□

□□

□□□O□□□□□□□□□□□□□□□□□

1.3 □□□□□□□□□□□□□□□□

1.4 □□□□□□□□□□□□□□□□□□□□□□□□□□□□□□

1.5 □□□□□□□□□□□□□□□

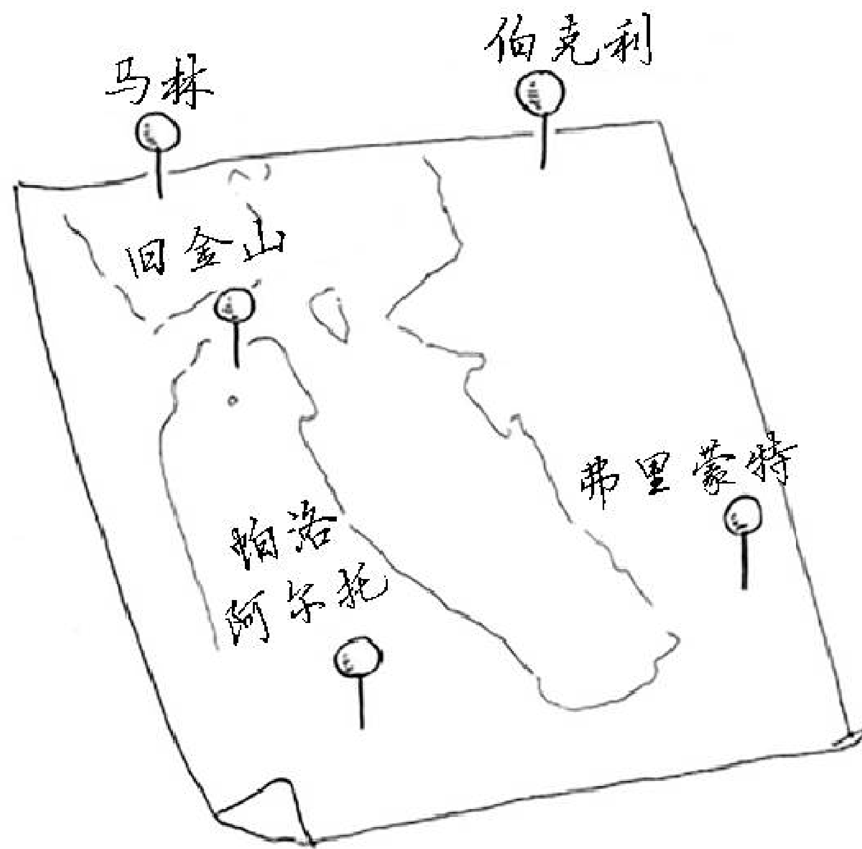
1.6 □□□□□□□□□A□□□□□□□□□□□□□□□□□□□□□□□4□□□ □□□□□□□□□□□□□

1.3.5 □□□

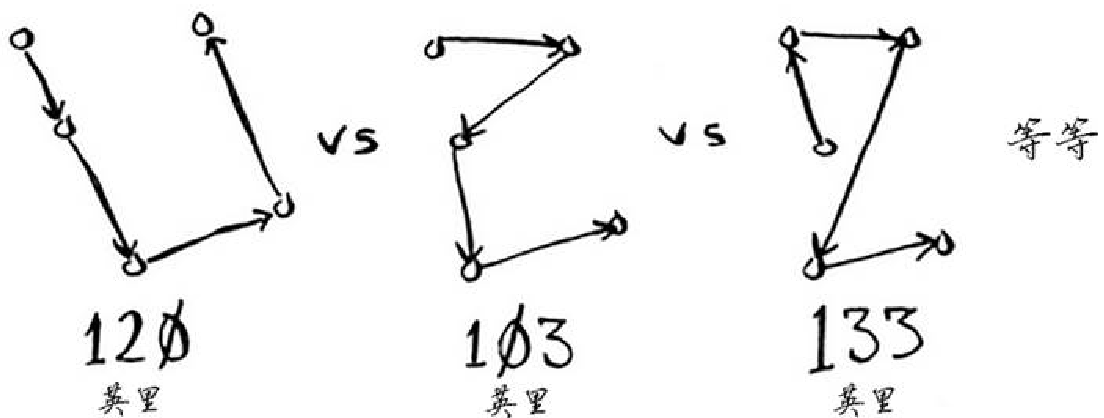
□□□□□□□□□□□□□□□□□□□□□□$O(n!)$□□

□□□□□□□

□□□□□5□□□□



□□□□□□□□□□□□Opus□□□□□□5□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□
□□□□□



□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□5□□□□120□□□□□□□□□□□□□□□□5
□□□□□□□□□□□□□□□□120□□□□□□6□□□□□□□□□□720□□□□□720□□□□□
□□□□□□□7□□□□□□□□□□5040□□□□

城市数	操作数
6	720
7	5040
8	40320
...	...
15	1,307,674,368,000
...	...
30	265,252,859,812,191,058,636,308,480,000,000

操作数激增

□□□□□□□n□□□□□□□□□$n!$□n□□□□□□□□□□□□□□□□□□□□□□$O(n!)$□□□100□□□□□□□□□□□□□□□□□□——□□□□□□□□□□□□□□

□□□□□□□□Opus□□□10□□

□□

1.4 □□

- □□□□□□□□□□□□□□□□
- $O(\log n)$□$O(n)$□□□□□□□□□□□□□□□□□□□□□□□
- □□□□□□□□□□□□□□
- □□□□□□□□□□□□□□□□□□
- □□□□□□□□O□□□□□□

[illegible] 2 [illegible] [illegible]

[illegible]

- [illegible]——[illegible]1[illegible]
- [illegible]

[illegible]

[illegible]O[illegible]1[illegible]O[illegible]

2.1 [illegible]

[illegible]

麻烦来两个抽屉
先生，你可以使用这两个抽屉

fe0ffeeb

[illegible]——[illegible]

2.2 [illegible]

[illegible]

[illegible]

你的待办
事项清单
他人占用的
内存
吃午饭
玩地
滚球
喝茶
可用内存

[illegible]

这个地方已被占用，
无法在这里添加待办
事项
吃午饭
玩地
滚球
喝茶

[illegible] 4 [illegible]

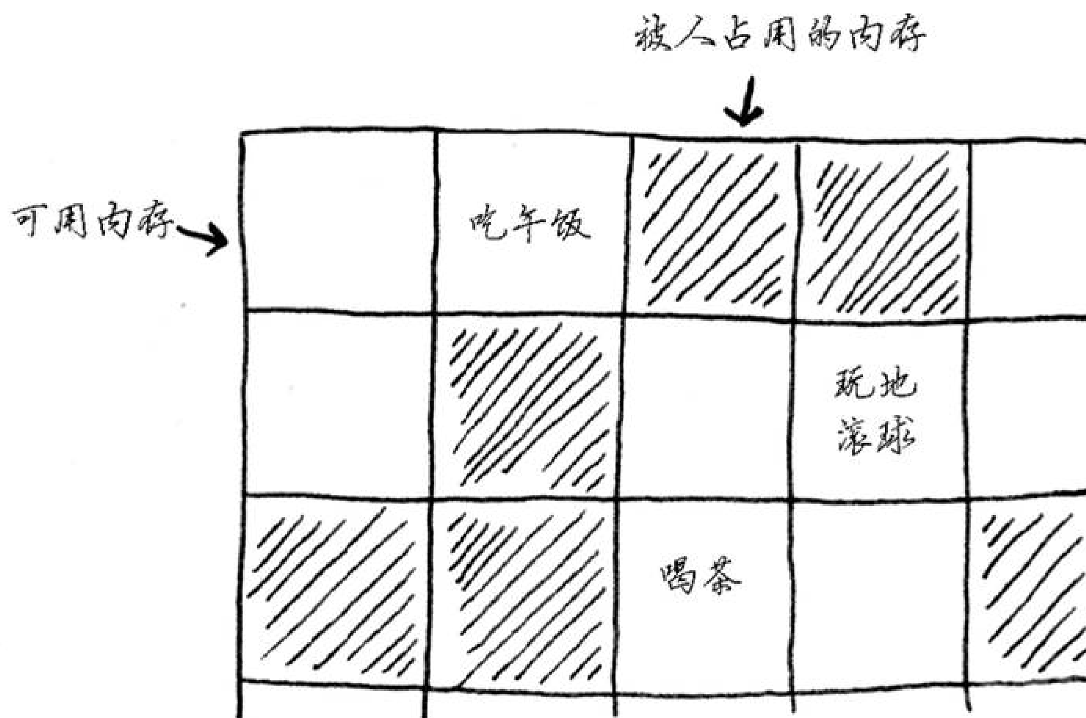
[illegible]

- [illegible]
- [illegible]

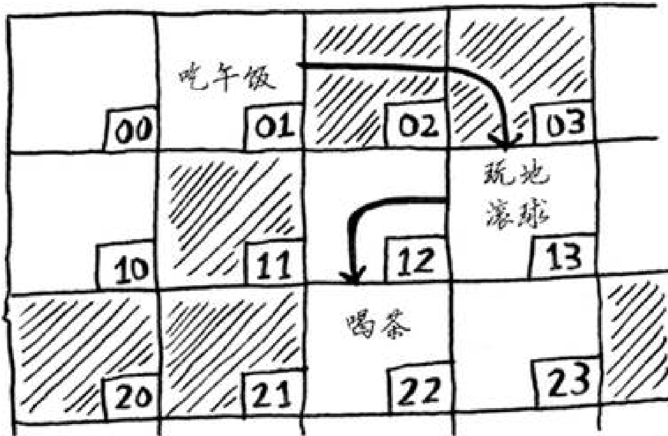
[illegible]

2.2.1 [illegible]

[illegible]



[illegible]



串在一起的内存地址

[illegible]“[illegible]123”[illegible]123[illegible]“[illegible]847”[illegible]

[illegible]10 000[illegible]10 000[illegible]“[illegible]”[illegible]

[illegible]

2.2.2 [illegible]

[illegible]Next[illegible]Newman[illegible]Next[illegible]Gustavo Fring[illegible]10[illegible]Next [illegible]

EVIL CAT NEXT

[illegible] #1 [illegible] #2 [illegible] #2 [illegible] #3 [illegible]

[illegible] 00 [illegible] #5 [illegible]

包含五个元素的数组

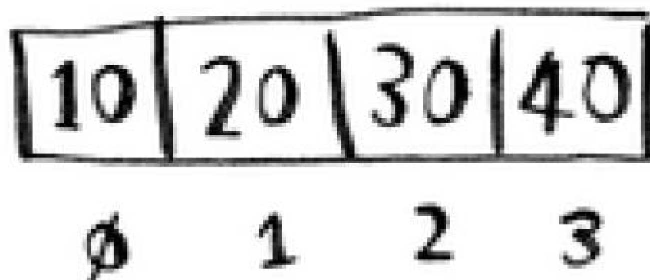
00 01 02 03 04

第五个元素

[illegible] 04 [illegible]

2.2.3 □□

□□□□□□□□□□□□0□□□1□□□□□□□□□□□□□□□□20□□□□1□



□□□10□□□□0□□□□□□□□□□□□□□0□□□□□□□□□□□□□□□□□□□□□□□□□□
□□□□□□□□□□□□□□□□0□□□□□□□□□□□□□□□□□□□□□□□□

□□□□□□□□□ □□□□□□“□□20□□□□1”□□□“□□20□□□□1□”□□□□□□□□□□
□□□□

□□□□□□□□□□□□□□□□□□□□□

	数组	链表
读取	O(1)	O(n)
插入	O(n)	O(1)

O(n) = 线性时间
O(1) = 常量时间

[illegible] $O(n)$ [illegible]

[illegible]

2.1 [illegible]

1. 买杂货

2. 看电影

3. SFBC会费

[illegible]

2.2.4 [illegible]

[illegible]

[illegible]

无序

有序

[illegible]

[illegible]

[illegible]

2.2.5 [illegible]

[illegible]

[illegible]

[illegible]

	数组	链表
读取	O(1)	O(n)
插入	O(n)	O(1)
删除	O(n)	O(1)

[illegible] $O(1)$ [illegible] $O(1)$ [illegible]

[illegible]

[illegible]

2.2 [illegible]

[illegible]

2.3 [illegible]Facebook[illegible]Facebook[illegible]Facebook[illegible]Facebook[illegible]——[illegible]

2.4 [illegible]Facebook[illegible]

2.5 [illegible]Facebook[illegible]Facebook[illegible]26[illegible]A[illegible]B[illegible]

[illegible]Adit B[illegible]Facebook[illegible]Adit B[illegible]Zakhir H[illegible]

26[illegible]z[illegible]Zakhir H[illegible]

[illegible]O[illegible]

2.3 [illegible]

[illegible]——[illegible]O[illegible]

[illegible]

	播放次数
RADIOHEAD	156
KISHORE KUMAR	141
THE BLACK KEYS	35
NEUTRAL MILK HOTEL	94
BECK	88
THE STROKES	61
WILCO	111

	播放次数
RADIOHEAD	156
KISHORE KUMAR	141
THE BLACK KEYS	35
NEUTRAL MILK HOTEL	94
BECK	88
THE STROKES	61
WILCO	111

1. RADIOHEAD是播放次数最多的乐队

→

排序后	播放次数
RADIOHEAD	156

2. 将其加入到新列表中

	播放次数
KISHORE KUMAR	141
THE BLACK KEYS	35
NEUTRAL MILK HOTEL	94
BECK	88
THE STROKES	61
WILCO	111

1. KISHORE KUMAR是播放次数第二多的乐队

→

排序后	播放次数
RADIOHEAD	156
KISHORE KUMAR	141

2. 因此它是下一个要加入到新列表中的乐队

♫	播放次数
RADIOHEAD	156
KISHORE KUMAR	141
WILCO	111
NEUTRAL MILK HOTEL	94
BECK	88
THE STROKES	61
THE BLACK KEYS	35

[illegible] $O(n)$ [illegible]

1. RADIOHEAD
2. KISHORE KUMAR
3. THE BLACK KEYS
4. NEUTRAL MILK HOTEL
5. BECK
6. THE STROKES
7. WILCO

n个元素

[illegible] $O(n)$ [illegible] $O(n)$ [illegible] n [illegible]

1. RADIOHEAD
2. KISHORE KUMAR
3. THE BLACK KEYS
4. NEUTRAL MILK HOTEL
5. BECK
6. THE STROKES
7. WILCO

$O(n)$

1. KISHORE KUMAR
2. THE BLACK KEYS
3. NEUTRAL MILK HOTEL
4. BECK
5. THE STROKES
6. WILCO

$O(n)$

...

1. THE STROKES

$O(n)$

n次

[illegible] $O(n \times n)$ [illegible] $O(n^2)$ [illegible]

[illegible]

- [illegible]
- [illegible]
- [illegible]

[illegible]

[illegible]$O(n^2)$[illegible]O[illegible]4[illegible]

[illegible]n[illegible]n[illegible]n - 1, n – 2, …, 2[illegible]1[illegible]1/2 × n[illegible]$O(n \times 1/2 \times n)$[illegible]O[illegible]1/2[illegible]4[illegible]$O(n \times n)$[illegible]$O(n^2)$[illegible]

[illegible]$O(n \log n)$[illegible]

[illegible]

[illegible]

```
def findSmallest(arr):
  smallest = arr[0]  ←-------[illegible]
  smallest_index = 0  ←-------[illegible]
  for i in range(1, len(arr)):
    if arr[i] < smallest:
      smallest = arr[i]
      smallest_index = i
  return smallest_index
```

[illegible]

```
def selectionSort(arr):  ←------[illegible]
  newArr = []
  for i in range(len(arr)):
      smallest = findSmallest(arr)  ←------[illegible]
      newArr.append(arr.pop(smallest))
  return newArr

print selectionSort([5, 3, 6, 2, 10])
```

2.4 [illegible]

- [illegible]
- [illegible]
- [illegible]
- [illegible]

- [illegible]
- [illegible]
- [illegible]int[illegible]double[illegible]

[illegible] 3 [illegible] [illegible]

[illegible]

- [illegible]
- [illegible]4[illegible]

[illegible] ——[illegible]

- [illegible]
- [illegible]5[illegible]`factorial` [illegible]4[illegible]`factorial` [illegible]5[illegible]

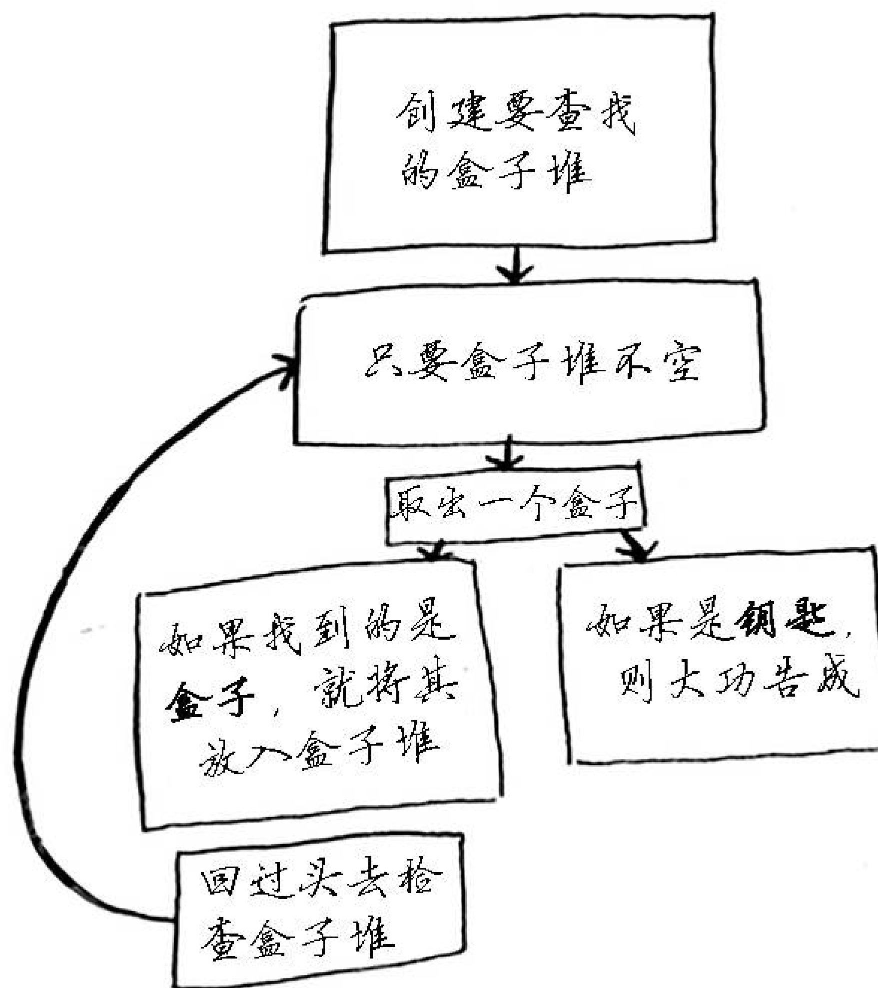
[illegible]

3.1 [illegible]

[illegible]

盒子里
的盒子

最外面
的盒子


创建要查找
的盒子堆
只要盒子堆不空
取出一个盒子
如果找到的是
盒子，就将其
放入盒子堆
如果是钥匙，
则大功告成
回过头去检
查盒子堆

(1) [illegible]

(2) [illegible]

(3) [illegible]

(4) [illegible]

(5) [illegible]

[illegible]

详细检查盒子内的每样东西

如果是盒子

如果是钥匙，则大功告成

(1) [illegible]

(2) [illegible]

(3) [illegible]

[illegible]while [illegible]

```
def look_for_key(main_box):
  pile = main_box.make_a_pile_to_look_through()
  while pile is not empty:
    box = pile.grab_a_box()
    for item in box:
      if item.is_a_box():
        pile.append(item)
      elif item.is_a_key():
        print "found the key!"
```

[illegible]——[illegible]

```
def look_for_key(box):
  for item in box:
```

```
    if item.is_a_box():
      look_for_key(item)  ←------[illegible]
    elif item.is_a_key():
      print "found the key!"
```

[illegible]Leigh Caldwell[illegible]Stack Overflow[illegible]“[illegible]”[1]

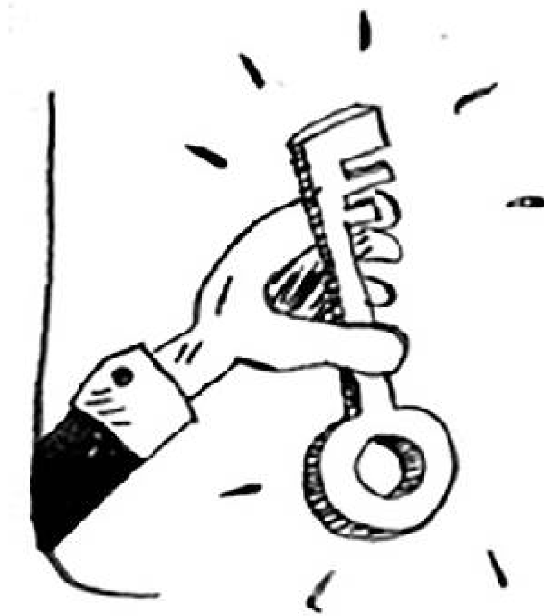
[1] [illegible]http://stackoverflow.com/a/72694/139117 [illegible]

[illegible]

3.2 [illegible]

[illegible]

```
> 3...2...1
```



□□□□□□□□□□□□□□□□□□□□

```
def countdown(i):
print i
countdown(i-1)
```

□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□



无限循环

```
> 3...2...1...0...-1...-2...
```

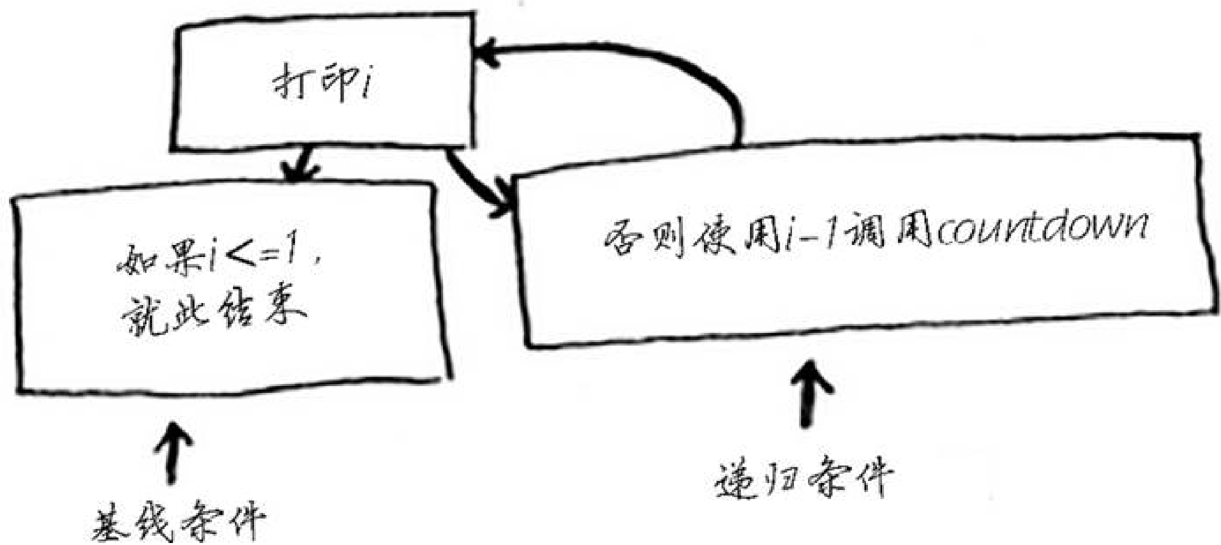
□□□□□□□□□□□□Ctrl+C□□

□□ □base case□□□□□□ □recursive case□□□□□□□□□□□□□□□□□□□□□□□□□□□ □□□□□□□□□□□□□□□□

□□□□□□countdown □□□□□□

```
def countdown(i):
    print i
  if i <= 0:  ←------□□□□
    return
  else:  ←------□□□□
    countdown(i-1)
```

□□□□□□□□□□□□□□□□□□□□□



3.3 □

□□□□□□□□□□□□□——□□□ □call stack□□□□□□□□□□□□□□□□□□□□ □□□□□□□□□

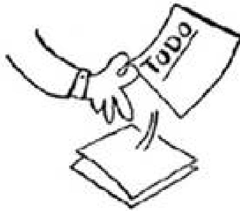
[illegible]——[illegible]

[illegible]





[illegible]











[illegible]

3.3.1 [illegible]

[illegible]

```
def greet(name):
    print "hello, " + name + "!"
    greet2(name)
    print "getting ready to say bye..."
    bye()
```

[illegible]

```
def greet2(name):
    print "how are you, " + name + "?"
def bye():
    print "ok bye!"
```

[illegible]

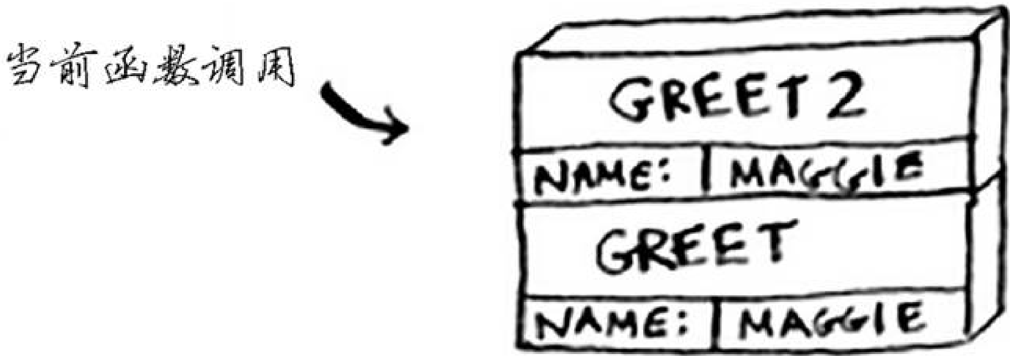
[illegible]

[illegible]Python[illegible]print [illegible]

[illegible]greet("maggie") [illegible]

[illegible]name [illegible]maggie [illegible]

hello, maggie! greet2("maggie")



how are you, maggie?



greet greet greet2 greet greet2 greet getting ready to say bye… bye



□□□□□□□□bye □□□□□□□□□□□ok bye! □□□□□□□□□□



□□□□□□□□□greet □□□□□□□□□□□□□□□□□greet □□□□□□□□□□□□□□
□□□□□□□□□□ □

□□

3.1 □□□□□□□□□□□□□□□□□□

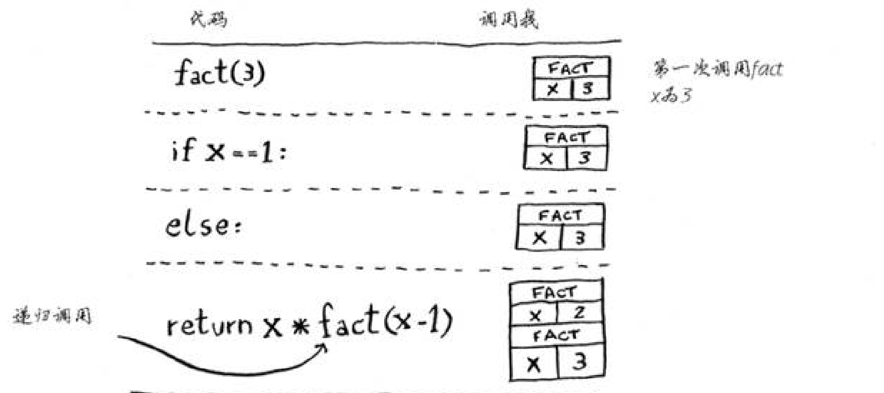
[illegible]

3.3.2 [illegible]

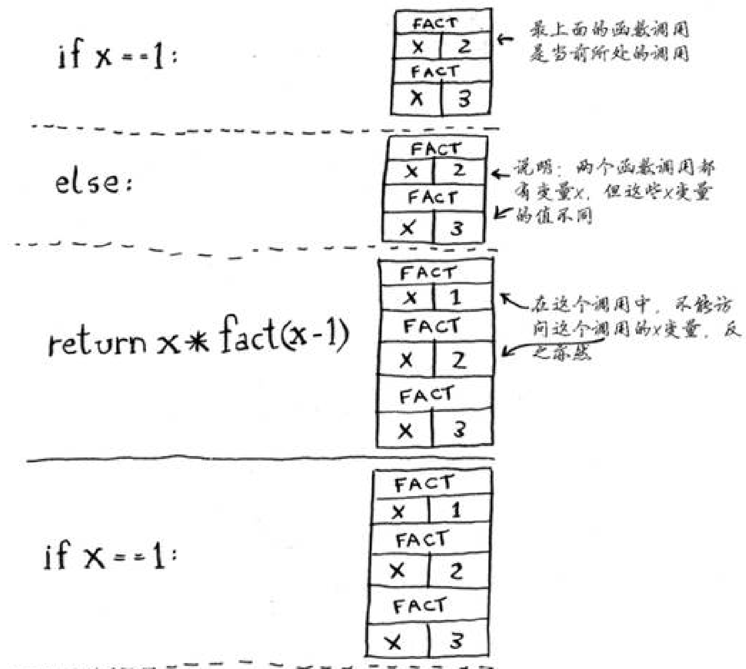
[illegible]factorial [illegible]factorial(5) [illegible]5![illegible]5! = 5 * 4 * 3 * 2 * 1[illegible]factorial(3) [illegible]3 * 2 * 1[illegible]

```
def fact(x):
  if x == 1:
    return 1
  else:
    return x * fact(x-1)
```

[illegible]fact(3) [illegible]

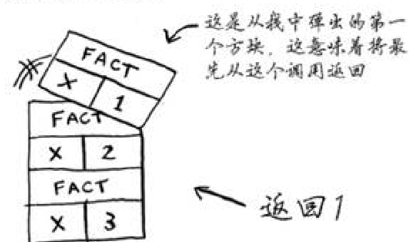


现在位于对fact的第二次调用中。x为2



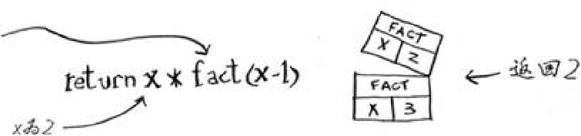
现在有三个对fact的调用，但还没有一个调用结束

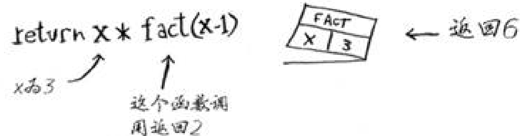
返回1



这是从栈中弹出的第一个方块，这意味着将最先从这个调用返回

返回1





□□□□□fact□□□□□□□x□□□□□□□□□□□□□□□□□□□x□□□

□□□

创建要查找的盒子堆
只要盒子堆不空
取出一个盒子
如果找到的是**盒子**，就将其放入盒子堆
如果是**钥匙**，则大功告成
回过头去检查盒子堆



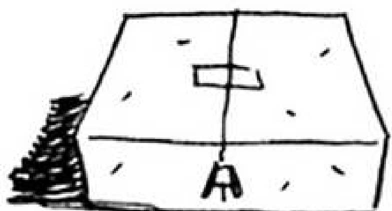
□□□

接下来要查找的盒子
盒子堆

[illegible]

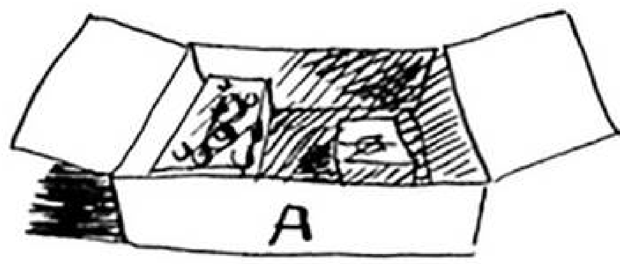
详细检查盒子内的每样东西
如果是盒子
如果是钥匙，则大功告成

[illegible]

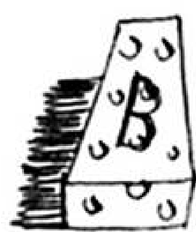


你仔细检查盒子A





在这个盒子中你发现了盒子B和C

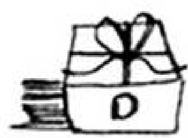


你检查盒子B



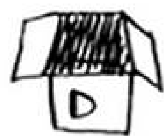


在其中发现了盒子D

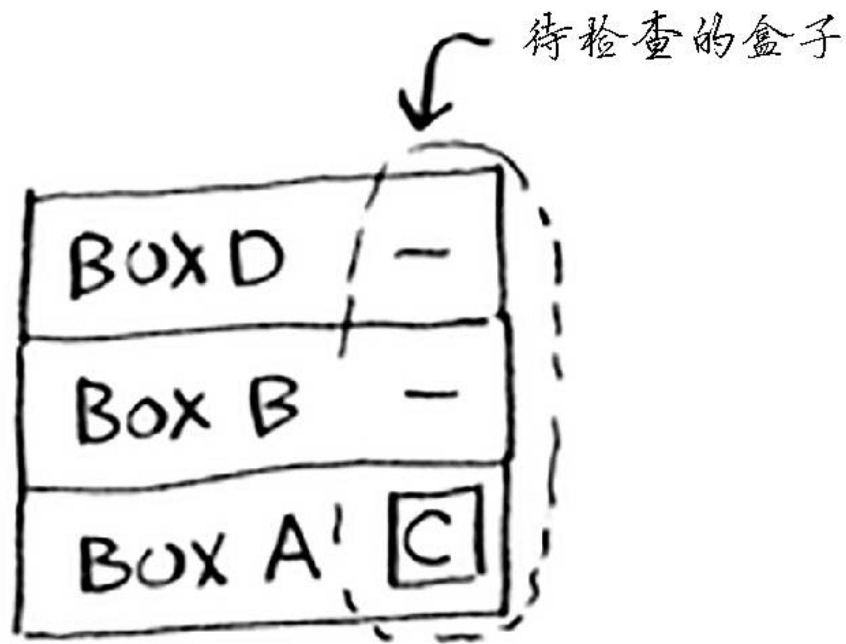


你检查盒子D





发现它是空的



3.2

3.4

- [illegible]
- [illegible]



[illegible] 4 [illegible]

[illegible]

- [illegible]
- [illegible]——[illegible]

[illegible]（divide and conquer，D&C）——[illegible]

[illegible]D&C[illegible]“[illegible]”

[illegible]D&C[illegible]——[illegible]2[illegible]

4.1 [illegible]

D&C[illegible]——[illegible]D&C[illegible]

[illegible]

1680 m

640 m

[illegible]

分出来的不是方块

分得太小

大小不同

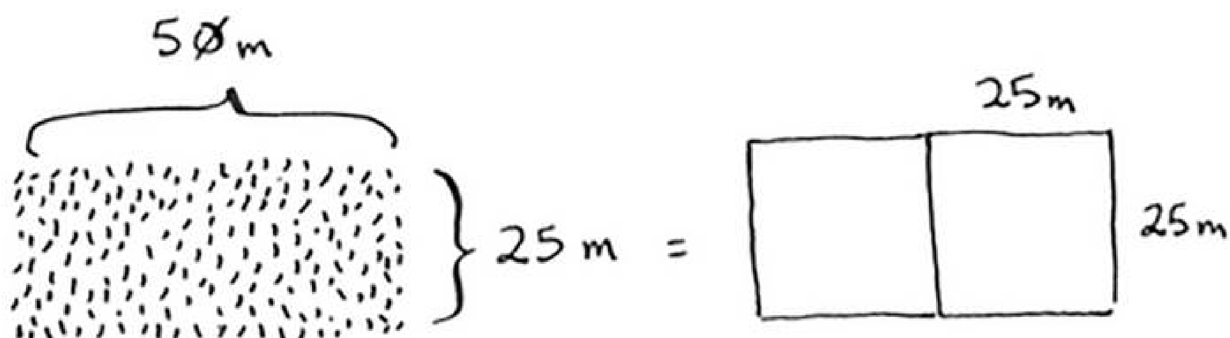
[illegible]D&C[illegible]D&C[illegible]
D&C[illegible]

(1) [illegible]

(2) [illegible]

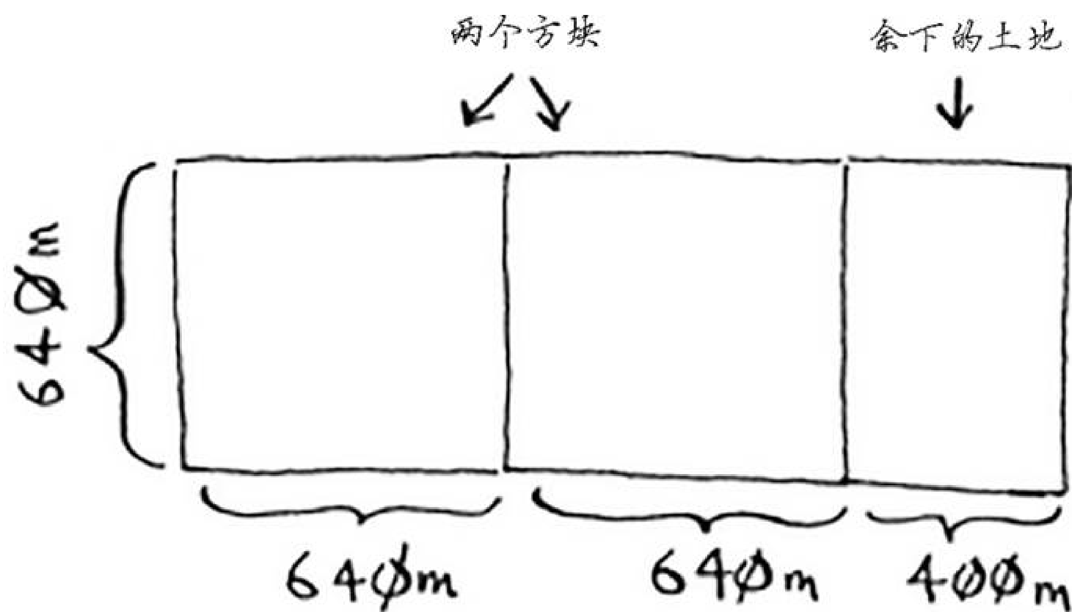
[illegible]D&C[illegible]

[illegible]



□□□□□25 m□□□□□50 m□□□□□□□□□□□□ 25 m×25 m□□□□□□□□□□
□□□□□□□□□□□

□□□□□□□□□□□□□□D&C□□□□□□□□D&C□□□□□□□□□□□□□□□□□□□□□□
□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□



□□□□□□□□□□□□640 m×640 m□□□□□□□□□□□□□□□□□□□□□□□□□□□□
□□□□□□□□□□□□□□

[illegible]1680 m×640 m[illegible]640 m×400 m[illegible] [illegible]1680 m×640 m[illegible]640 m×400 m[illegible]

[illegible]

[illegible]“[illegible]”[illegible]

https://www.khanacademy.org/computing/computer-science/ryptography/modarithmetic/a/the-euclidean-algorithm [illegible]

[illegible]640 m × 400 m[illegible]400 m × 400 m[illegible]

[illegible]400 m × 240 m[illegible]

[illegible]240 m × 160 m[illegible]

[illegible]

[illegible]160[illegible]80[illegible]

[illegible]80 m× 80 m[illegible]

[illegible]D&C[illegible]

(1) [illegible]

(2) [illegible]

D&C[illegible]

[illegible]

[illegible]

```
def sum(arr):
  total = 0
  for x in arr:
    total += x
  return total

print sum([1, 2, 3, 4])
```

[illegible]

[illegible]

[illegible]

[illegible]

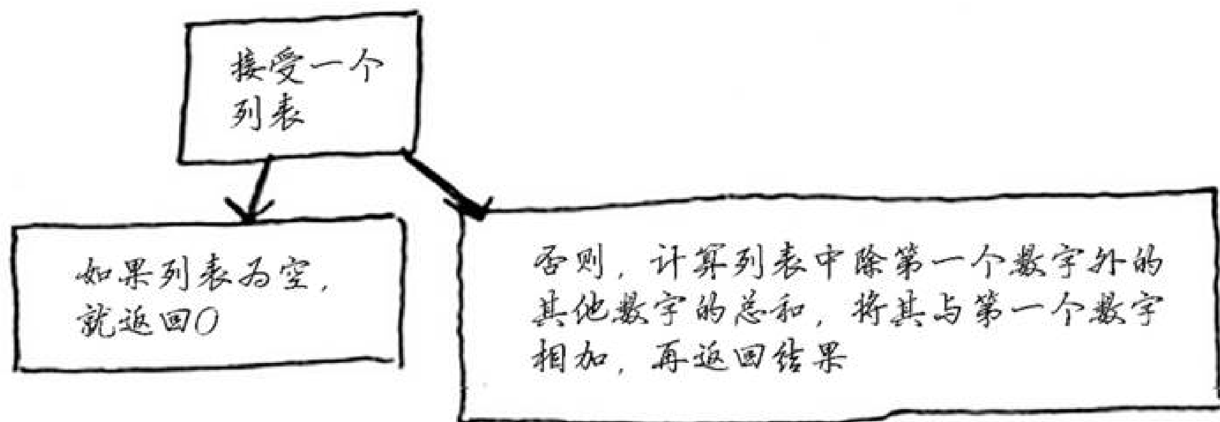
$$sum([2, 4, 6]) = 12$$

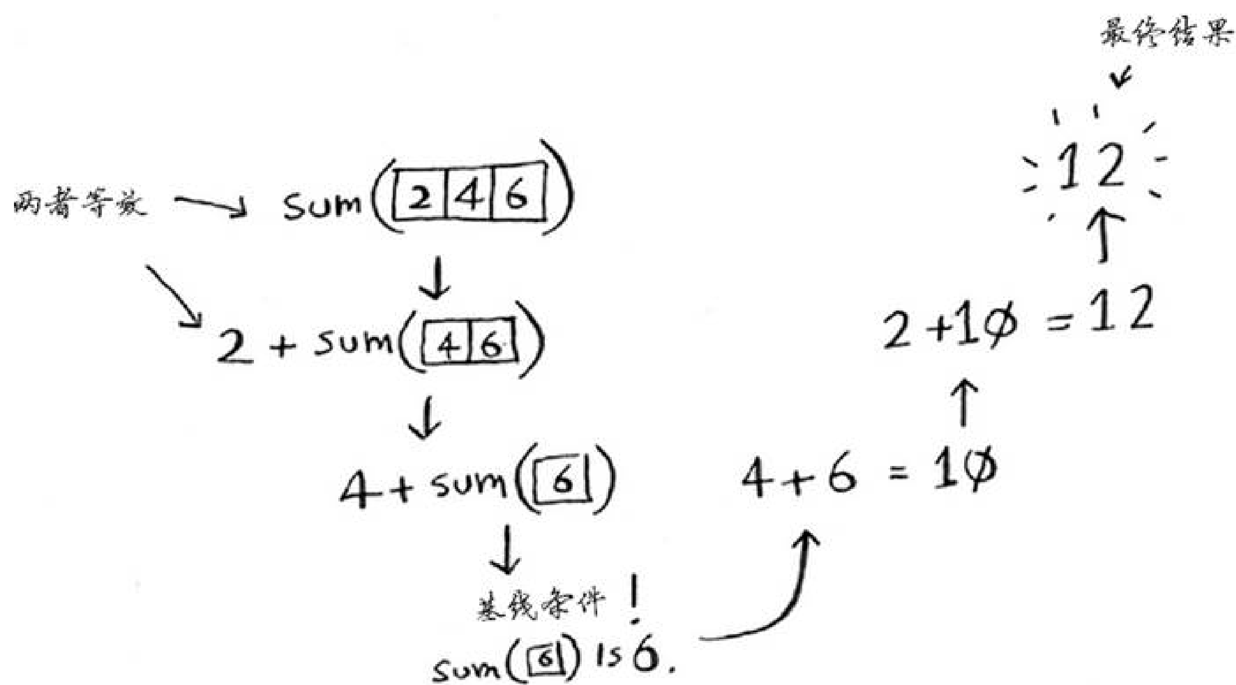
[illegible]

$$2 + sum([4, 6]) = 2 + 10 = 12$$

[illegible] 12 [illegible] sum [illegible]

[illegible] sum [illegible]


接受一个列表
如果列表为空，就返回0
否则，计算列表中除第一个数字外的其他数字的总和，将其与第一个数字相加，再返回结果


最终结果
12
两者等效
sum(2 4 6)
2 + sum(4 6)
4 + sum(6)
基线条件！
sum(6) is 6.
4 + 6 = 10
2 + 10 = 12

遇到基线条件前，这些函数调用都未完成！

sum([2|4|6])

2 + sum([4|6])

4 + sum([6])

基线条件！

sum([6]) is 6.

这个函数调用最先完成

别忘了，递归保存了这些未完成的函数调用的状态

4 + 6 = 10

2 + 10 = 12

12

[illegible]

[illegible]

[illegible]

[illegible] Haskell [illegible] Haskell [illegible] sum [illegible]

```
sum [] = 0  ←---------------------[illegible]
sum (x:xs) = x + (sum xs)  ←------[illegible]
```

[illegible]
[illegible]Haskell[illegible]if [illegible]

```
sum arr = if arr == []
            then 0
            else (head arr) + (sum (tail arr))
```

[illegible]Haskell[illegible]
[illegible]Haskell[illegible]

[illegible]

4.1 [illegible]sum [illegible]

4.2 [illegible]

4.3 [illegible]

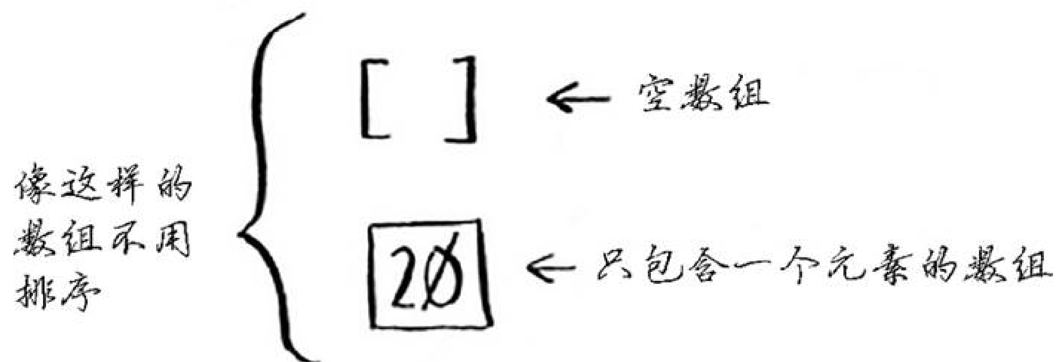
4.4 [illegible]1[illegible]
[illegible]



4.2 [illegible]

[illegible]C[illegible]qsort [illegible]D&C[illegible]

[illegible]“[illegible]”[illegible]



```
def quicksort(array):
  if len(array) < 2:
    return array
```

□□□□□□□□□□□

33 15 10

□□□□□□□□D&C□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□
□□□□□□□□□□□□□□□□□□□ □pivot□□

□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□

33

基准值

□□□□□□□□□□□□□□□□□□□□□□□□□

小于33的元素

大于33的元素
（空数组）

15 10

33

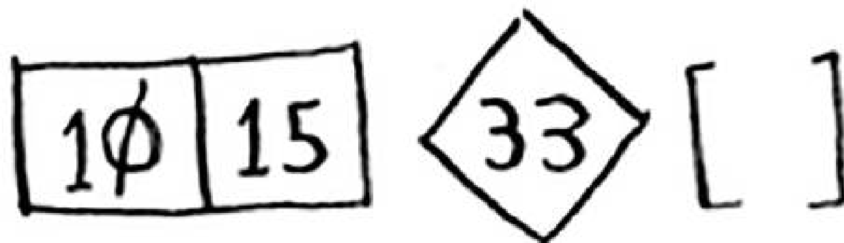
[ ]

基准值

□□□□□□ □partitioning□□□□□□□

- □□□□□□□□□□□□□□□□□□□□
- □□□□
- □□□□□□□□□□□□□□□□□□□□

□□□



□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□ + □□□ + □□□□□□□□
□□□□[10, 15] + [33] + []□□□□□□□□[10, 15, 33]□

□□□
□□□

```
quicksort([15, 10]) + [33] + quicksort([])
> [10, 15, 33]  ←-------□□□□□
```

□□□□□□□□□□□□□□□□□□□□□□15□□□□□□



□□□
□□□□□□□

(1) [illegible]

(2) [illegible]

(3) [illegible]

[illegible]

[illegible]33[illegible]

[illegible]

[illegible]

[illegible]

[illegible]0[illegible]4[illegible]0[illegible]4[illegible]

[illegible]3[illegible]

[illegible]5[illegible]

[illegible]

D&C

```
def quicksort(array):
  if len(array) < 2:
    return array  ←------
  else:
```

```
    pivot = array[0]  ←------□□□□
    less = [i for i in array[1:] if i <= pivot]  ←------□□□□□□□□□
□□□□□

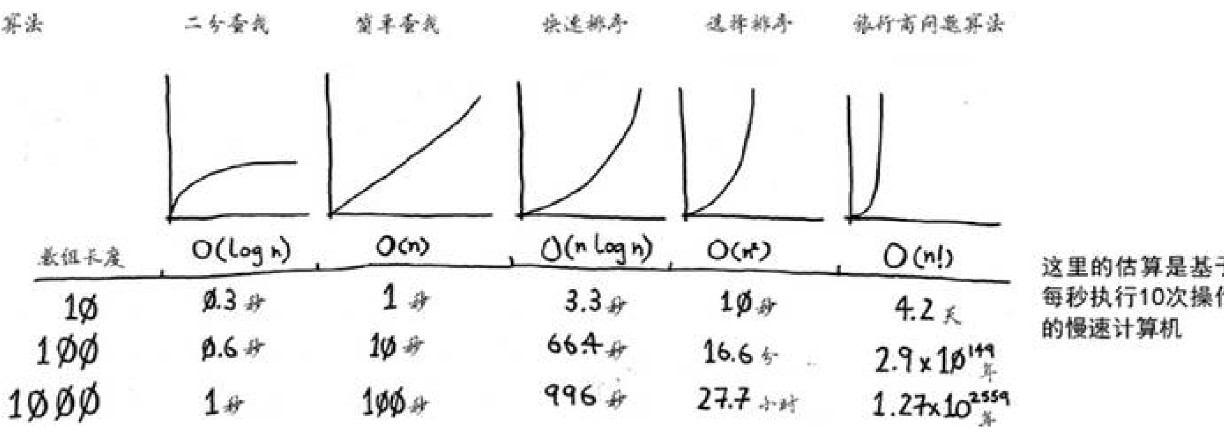
    greater = [i for i in array[1:] if i > pivot]  ←------□□□□□□□□
□□□□□□□

    return quicksort(less) + [pivot] + quicksort(greater)

print quicksort([10, 5, 2, 3])
```

4.3 □□□O□□□

□□□O□□□□□



□□□□□□□□□□□□□□□10□□10□□

□□□□□□□□□□□□□□□□□□□□□□□□□2□□□□□□□□□□□□□$O(n^2)$□□□□□□□

□□□□□□□□□□ □merge sort□□□□□□□□□□□□□$O(n \log n)$□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□$O(n^2)$□

[illegible]$O(n \log n)$[illegible]

- [illegible]
- [illegible]$O(n \log n)$[illegible]$O(n \log n)$[illegible]

4.3.1 [illegible]

[illegible]

```
def print_items(list):
  for item in list:
    print item
```

[illegible]$O(n)$[illegible]1[illegible]

```
from time import sleep
def print_items2(list):
  for item in list:
    sleep(1)
    print item
```

[illegible]1[illegible]5[illegible]

2 4 6 8 10

print_items: 2 4 6 8 10

print_items2: <休眠> 2 <休眠> 4 <休眠> 6 <休眠> 8 <休眠> 10

[illegible]$O(n)$[illegible]
`print_items` [illegible]1[illegible]O[illegible]
[illegible]`print_items` [illegible]O[illegible]$O(n)$[illegible]n [illegible]
[illegible]

c * n

固定的
时间量

c [illegible] [illegible]`print_items` [illegible]10[illegible] * n
[illegible]`print_items2` [illegible]1[illegible] * n [illegible]

[illegible]O[illegible]
[illegible]

10毫秒 * n	1秒 * log n
简单查找	二分查找

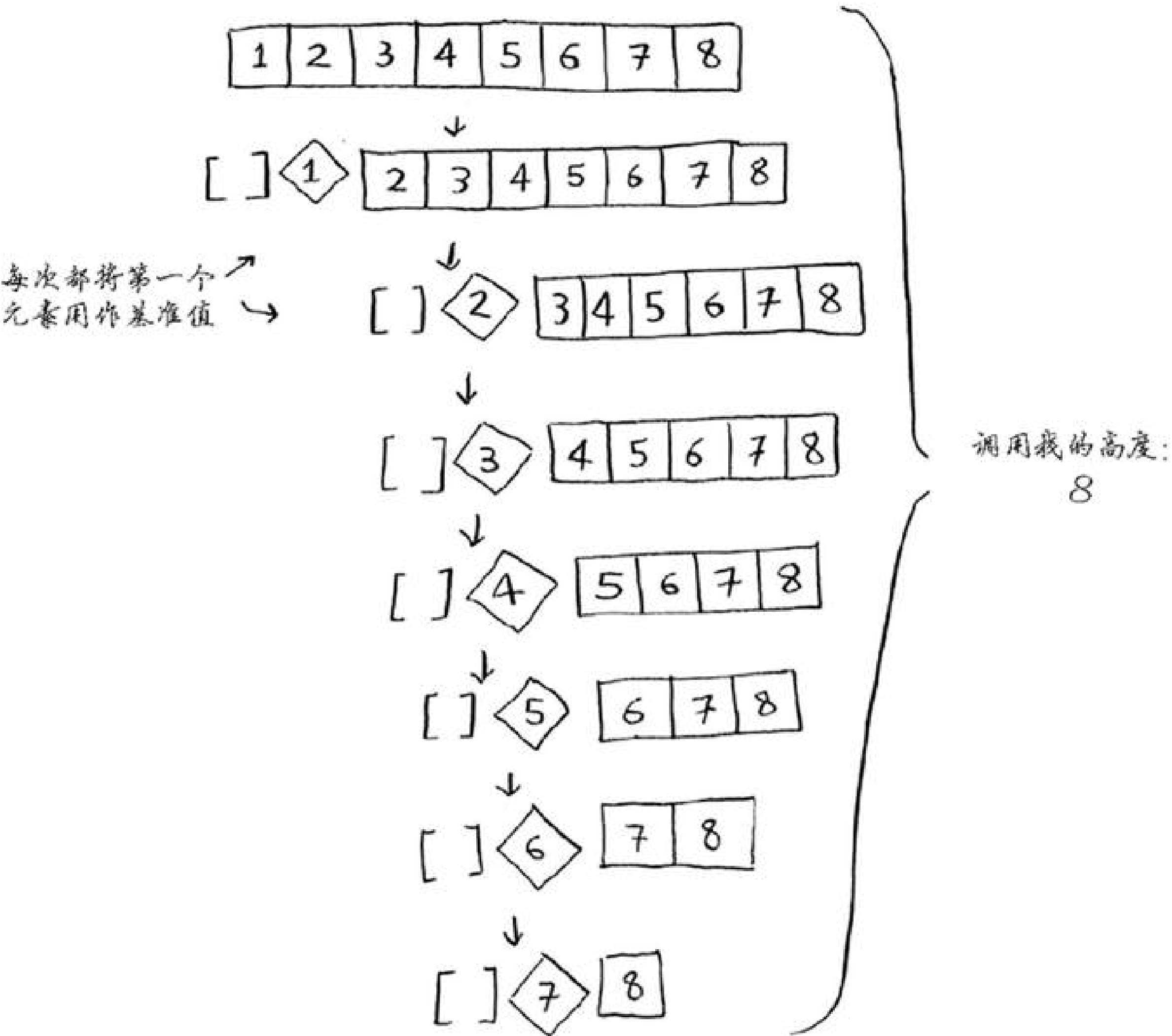
[illegible]10[illegible]1[illegible]
[illegible]40[illegible]

简单查找	10毫秒 * 40亿	= 463天
二分查找	1秒 * 32	= 32秒

[illegible]

$O(n \log n)$

4.3.2

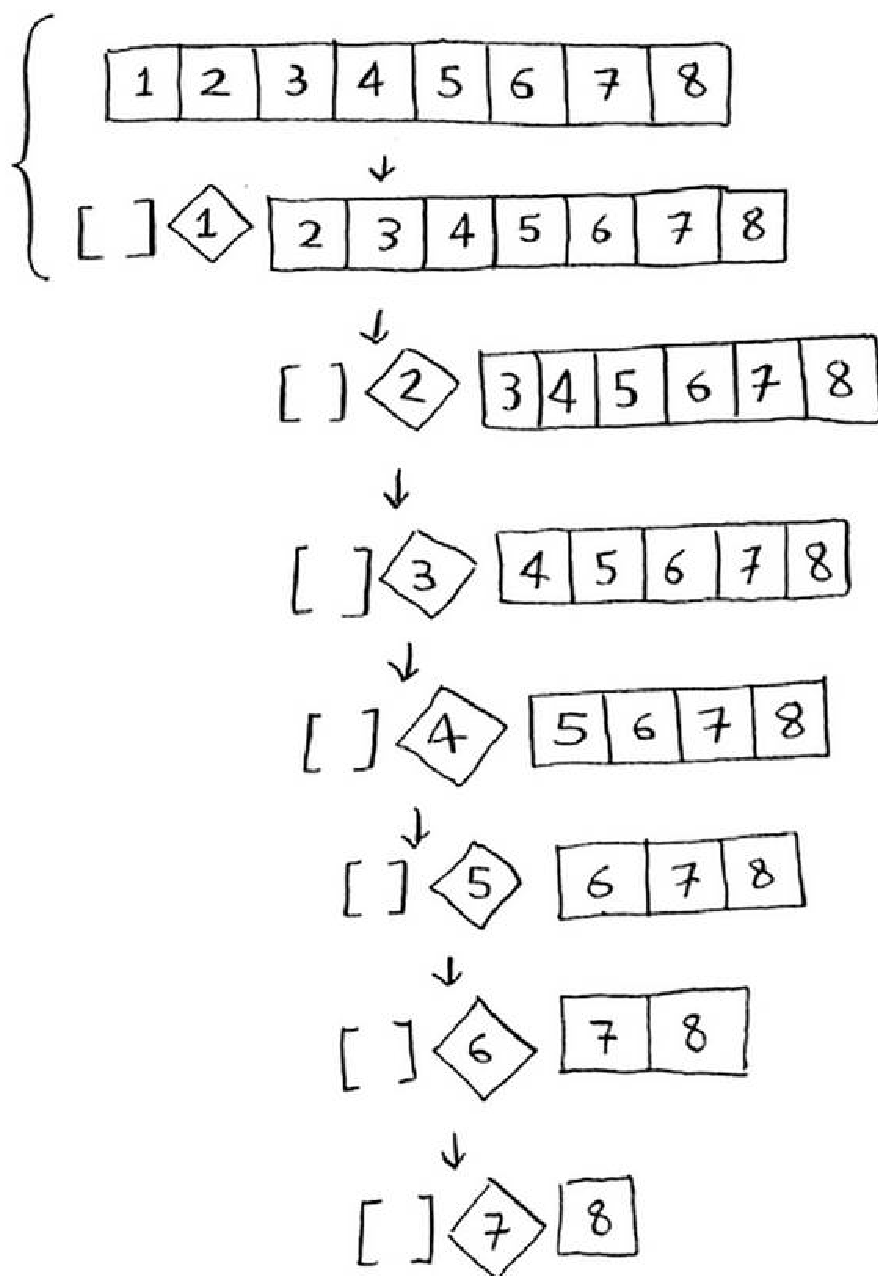


调用栈的长度：
4

[illegible]

[illegible] $O(n)$ [illegible] $O(\log n)$ [illegible]

[illegible] 8 [illegible] $O(n)$ [illegible] 8 [illegible] $O(n)$ [illegible]



[illegible]$O(n)$[illegible]

8个元素，即
O(n)个元素

8个元素，即
O(n)个元素

还是O(n)个
元素

□□□□□□□□□□□□□□□$O(n)$□

在这层完成
划分所需的
时间为O(n)

运行时间为
O(n)

运行时间为
O(n)

这层的运行时
间也为O(n)

层数：
O(log n)

□□□□□□□□□□□$O(\log n)$□□□□□□□□□□□□□□□□$O(\log n)$□□□□□□□□□□□□$O(n)$□□□□□□□□□□□□□□$O(n) * O(\log n) = O(n \log n)$□□□□□□□□□□

□□□□□□□□□$O(n)$□□□□□□□□□□□□□□$O(n) * O(n) = O(n^2)$□

□□$O(n \log n)$□□□□□□□□□□□□□□□□□□□D&C□□□

□□

□□□O□□□□□□□□□□□□□□□□□□□

4.5 □□□□□□□□□□□□

4.6 □□□□□□□□□□□□□2□

4.7 □□□□□□□□□□□□□□2□

4.8 □□□□□□□□□□□□□□□□□□□□□□□[2, 3, 7, 8, 10]□□□□□□□ □□
□2□□□□□□□□□□3□□□□□□□□□□□7□□□□□□

4.4 □□

- D&C□□□□□□□□□□D&C□□□□□□□□□□□□□□□□□□□□□□□□□□□□□
- □□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□$O(n \log n)$□
- □O□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□
- □□□□□□□□□□□□□□□□□□□□□□□□□□□□□□$O(\log n)$□□□□$O(n)$□
  □□□

[illegible] 5 [illegible] [illegible]

[illegible]

- [illegible]——[illegible]
- [illegible]



[illegible] [illegible]apple[illegible]1[illegible]$O(n)$[illegible]$O(\log n)$[illegible]

EGGS...2.49$
MILK....1.99$
PEAR....79¢

有序列表

O(log n)

列表长度

时间

PEAR..79¢
EGGS...2.49$
MILK....1.99$

无序列表

O(n)

列表长度

时间

[illegible] $O(n)$ [illegible] $O(\log n)$ [illegible] 10 [illegible]
[illegible]

本子中的商品数量	O(n)	O(log n)	
100	10秒	1秒	← 你需要检查 $\log_2 100 = 7$行
1000	1.66分	1秒	← 你需要检查 $\log_2 1000 = 10$行
10000	16.6分	2秒	↖ 你需要检查 $\log_2 10000 = 14$行

[illegible]

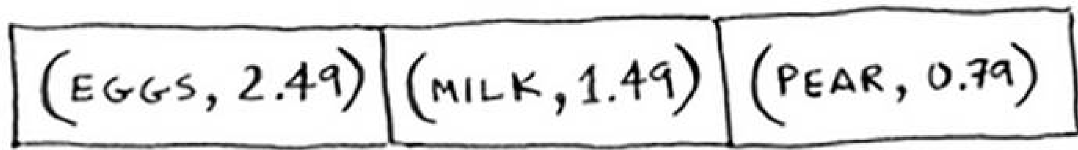


[illegible]Maggie[illegible]$O(1)$[illegible]

本子中的商品数量	简单查找 O(n)	二分查找 O(Log n)	MAGGIE O(1)
100	10秒	1秒	立即
1000	1.6分	1秒	立即
10000	16.6分	2秒	立即

[illegible]

[illegible]



[illegible] $O(\log n)$ [illegible] $O(1)$ [illegible] Maggie [illegible]

5.1 [illegible]

[illegible]

[illegible] “[illegible]” [illegible]

- [illegible] apple [illegible] 4 [illegible] apple [illegible] 4 [illegible]

- [illegible]1[illegible]

[illegible]“Maggie”[illegible]

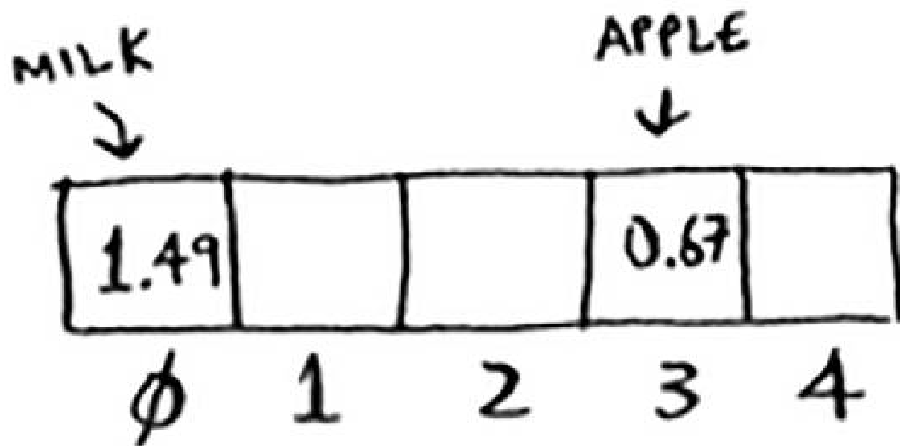
[illegible]

[illegible]apple[illegible]

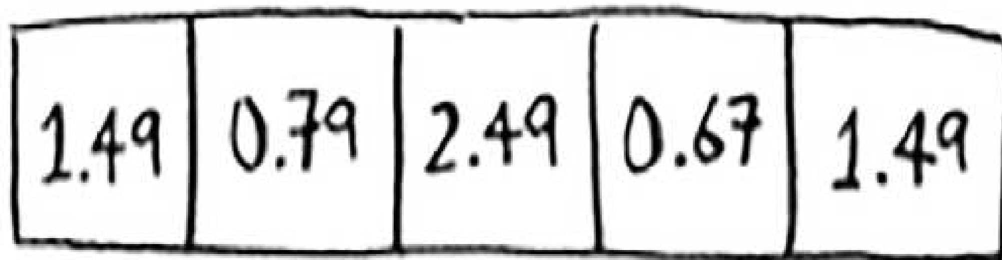
[illegible]3[illegible]3[illegible]

[illegible]milk[illegible]milk[illegible]

[illegible]0[illegible]0[illegible]

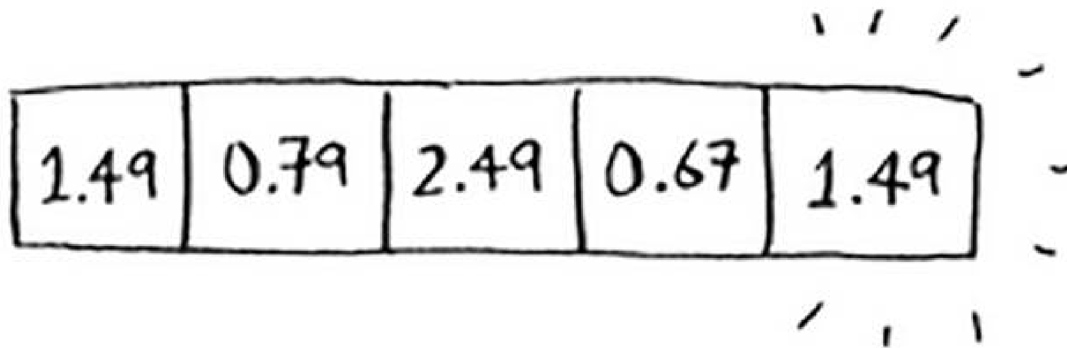


[illegible]



[illegible]avocado[illegible]avocado[illegible]

[illegible]4[illegible]



[illegible]

- [illegible]avocado[illegible]
- [illegible]avocado[illegible]4[illegible]milk[illegible]0[illegible]
- [illegible]5[illegible]100[illegible]

[illegible]“Maggie”[illegible] [illegible]hash table[illegible]

[illegible]2[illegible]

[illegible]Python[illegible]dict [illegible]

```
>>> book = dict()
```



□□□□□book □□□□□□□□□□□□□□□

```
>>> book["apple"] = 0.67  ←-------□□□□□□□67□□
>>> book["milk"] = 1.49  ←-------□□□□□1.49□□
>>> book["avocado"] = 1.49
>>> print book
{'avocado': 1.49, 'apple': 0.67, 'milk': 1.49}
```



□□□□□□□□□□□□□□□□

```
>>> print book["avocado"]
1.49  ←-------[illegible]
```

[illegible]book [illegible]

[illegible]

[illegible]

[illegible]

[illegible]

5.1 f(x) = 1 ←------[illegible]1

5.2 f(x) = rand() ←------[illegible]

5.3 f(x) = next_empty_slot() ←------[illegible]

5.4 f(x) = len(x) ←------[illegible]

5.2 [illegible]

5.2.1



BADE MAMA → 581 660 9820

ALEX MANNING → 484 234 4680

JANE MARIN → 415 567 3579

- [illegible]
- [illegible]

[illegible]

```
>>> phone_book = dict()
```

[illegible]Python[illegible]——[illegible]

```
>>> phone_book = {} ←------[illegible]phone_book = dict()[illegible]
```

[illegible]

```
>>> phone_book["jenny"] = 8675309
>>> phone_book["emergency"] = 911
```

[illegible]Jenny[illegible]

```
>>> print phone_book["jenny"]
8675309  ←------Jenny[illegible]
```

使用散列表创建的电话簿

[illegible]

[illegible]http://adit.io [illegible]
adit.io[illegible]IP[illegible]

ADIT.IO → 173.255.248.55

[illegible]IP[illegible]

GOOGLE.COM → 74.125.239.133
FACEBOOK.COM → 173.252.120.6
SCRIBD.COM → 23.235.47.175

[illegible]IP[illegible]**DNS**[illegible] [illegible]DNS
resolution[illegible]

5.2.2 [illegible]

VOTE

VOTERS
JIM
PAM
MIKE

□□□

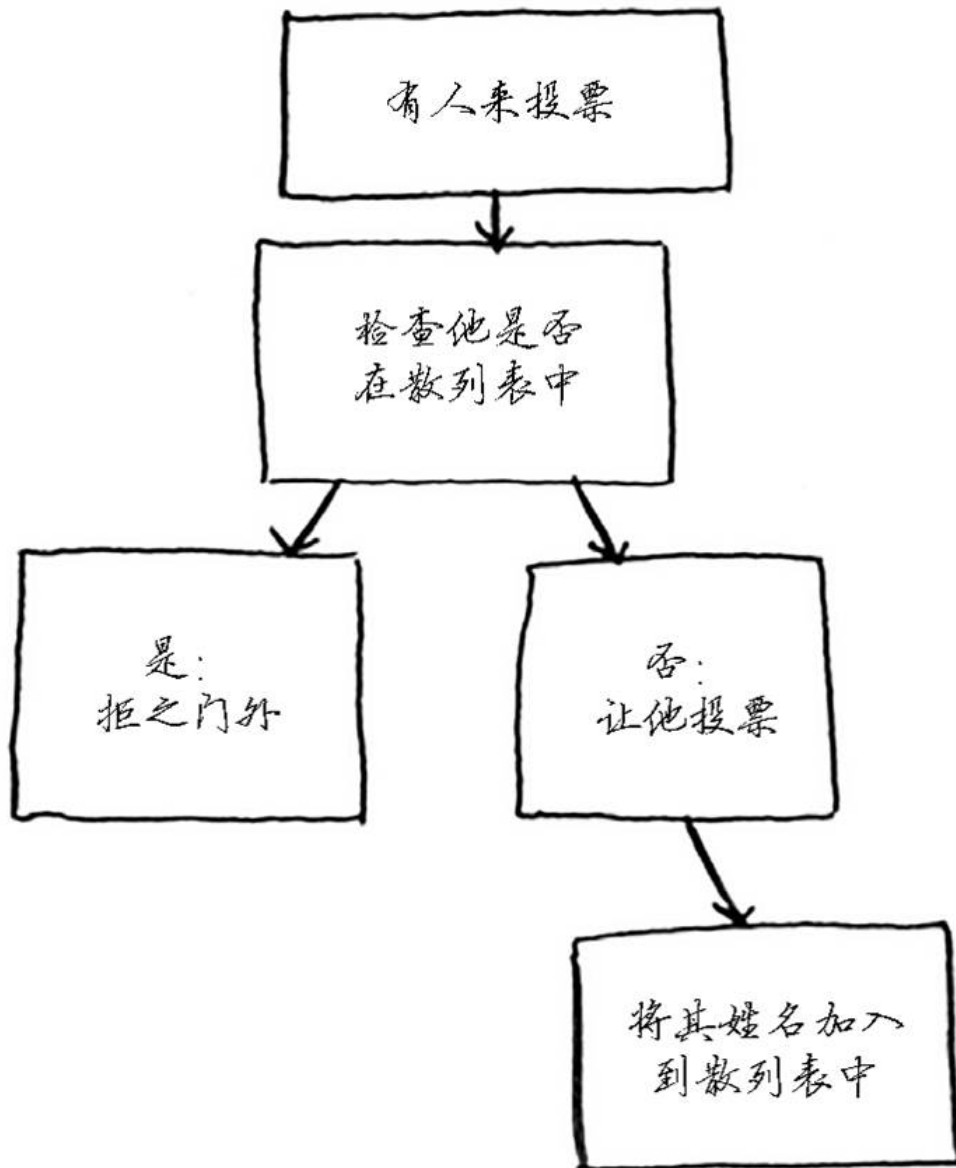
□□□□□□□□□□□□□□□□□□□□□□□

```
>>> voted = {}
```

□□□□□□□□□□□□□□□□□□

```
>>> value = voted.get("tom")
```

□□“tom”□□□□□□□□get □□□□□□□□□None □□□□□□□□□□□□□□□□□□□
□□□



□□□□□

```
voted = {}

def check_voter(name):
  if voted.get(name):
    print "kick them out!"
  else:
    voted[name] = True
    print "let them vote!"
```

□□□□□□□□

```
>>> check_voter("tom")
let them vote!
>>> check_voter("mike")
let them vote!
>>> check_voter("mike")
kick them out!
```

□□□□□□□Tom□□□□□□□`let them vote!`□□□Mike□□□□□□□□□`let them vote!`□□□□Mike□□□□□□□□□□□□`kick them out!`□

□□□

5.2.3 □□□□□□□□□

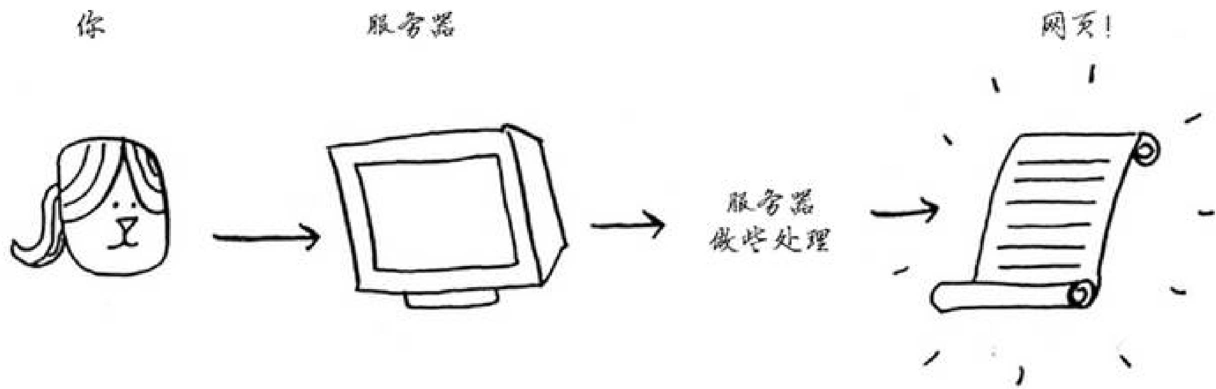
□□□facebook.com□



(1) □□Facebook□□□□□□□□□

(2) [illegible]

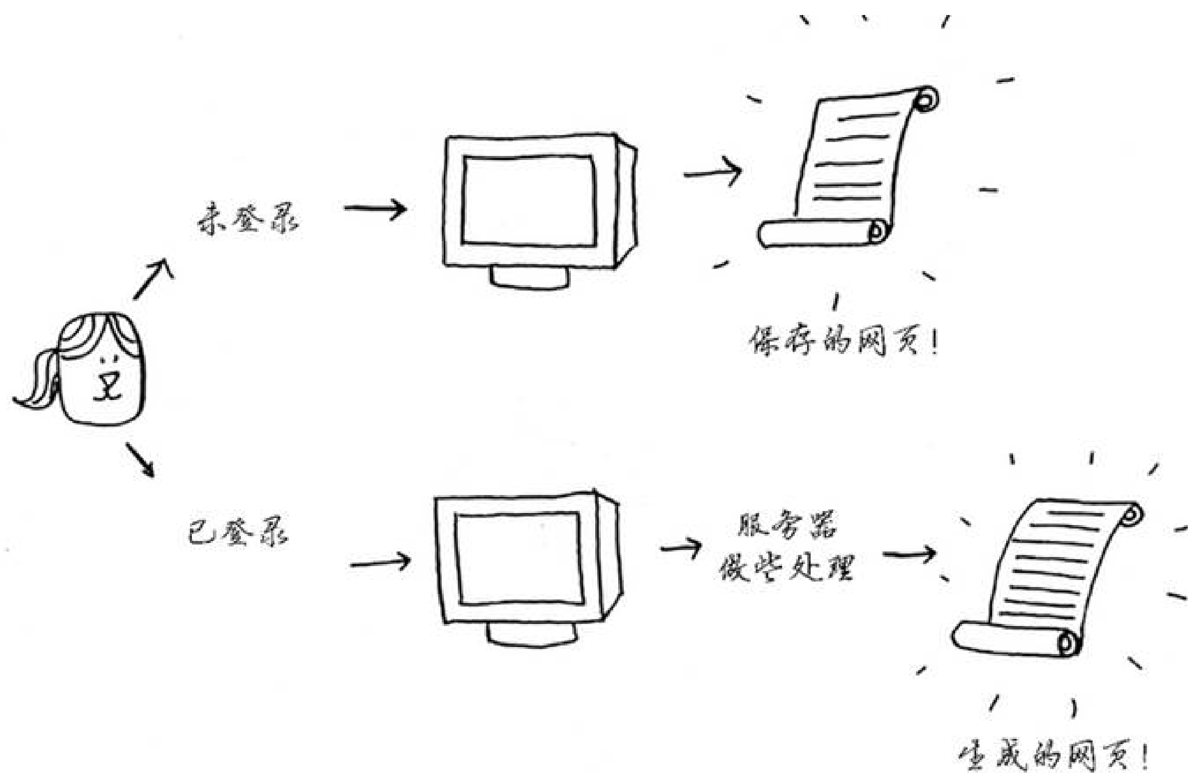
(3) [illegible]



[illegible]Facebook[illegible]Facebook[illegible]Facebook[illegible]Facebook[illegible]Facebook[illegible]Facebook[illegible]

[illegible]Google[illegible]238 900[illegible]Google[illegible]

[illegible]Facebook[illegible]facebook.com[illegible]Facebook[illegible]“[illegible]”[illegible]



[illegible] [illegible]

- [illegible]Google[illegible]

- Facebook[illegible]

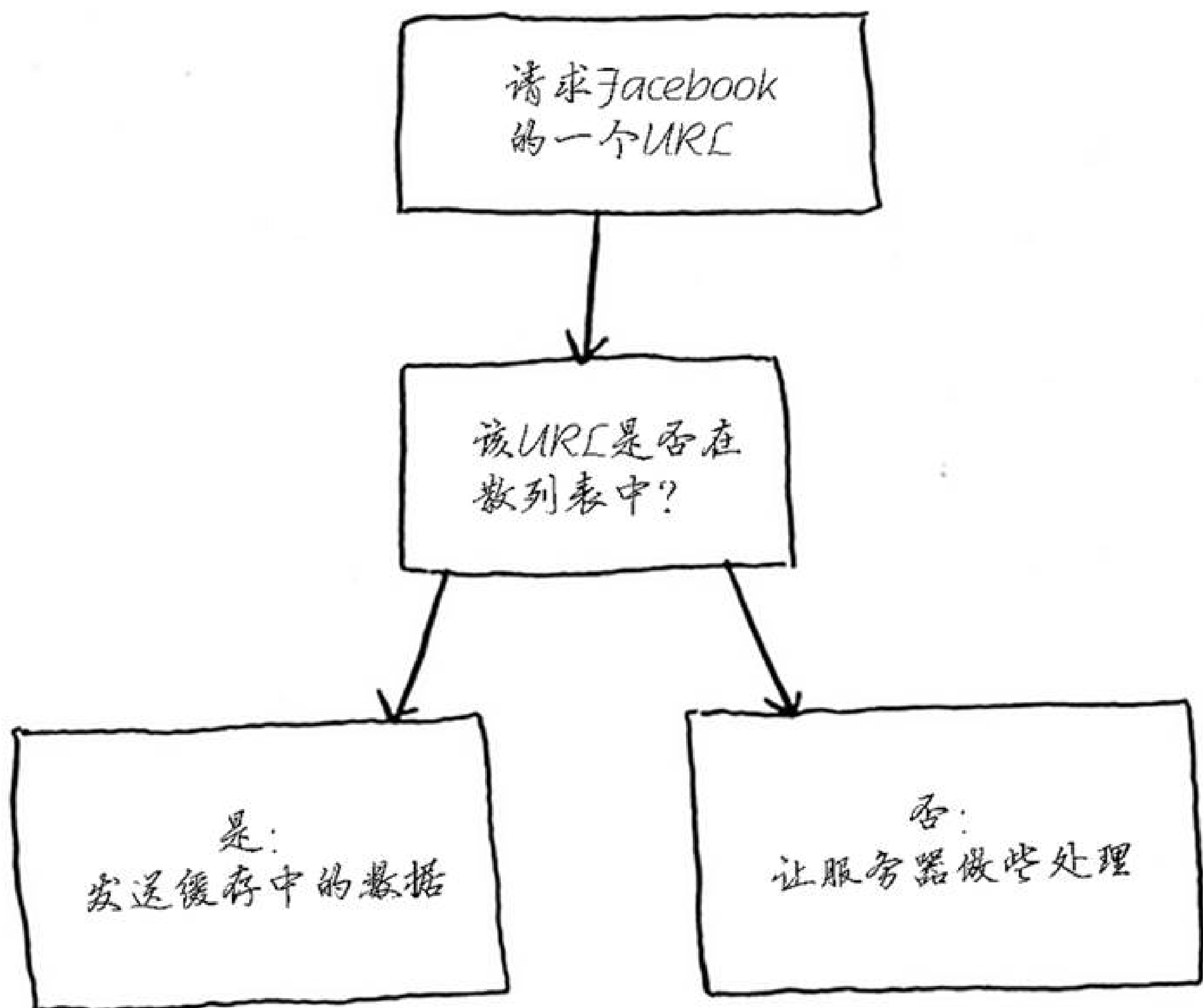
[illegible]

Facebook[illegible]About[illegible]Contact[illegible]Terms and Conditions[illegible]URL[illegible]

facebook.com/about → About页面的数据

facebook.com → 主页的数据

[illegible]Facebook[illegible]



□□□□□□□□

```
cache = {}

def get_page(url):
  if cache.get(url):
    return cache[url]  ←------□□□□□□□
  else:
    data = get_data_from_server(url)
    cache[url] = data  ←------□□□□□□□□□□
    return data
```

[illegible]URL[illegible]
[illegible]URL[illegible]

5.2.4 [illegible]

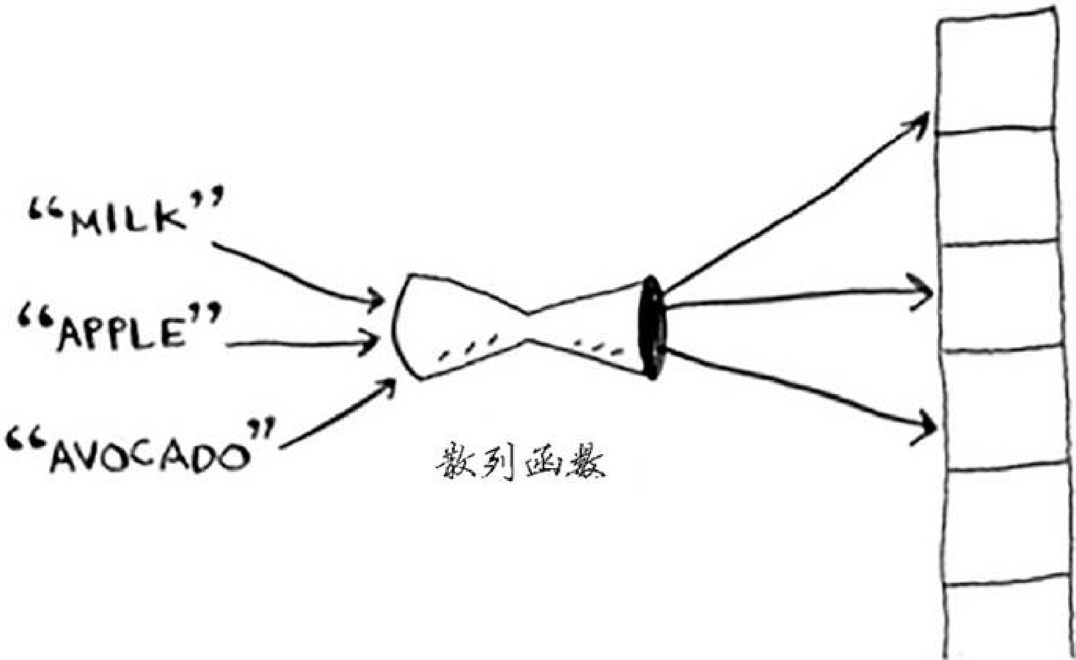
[illegible]

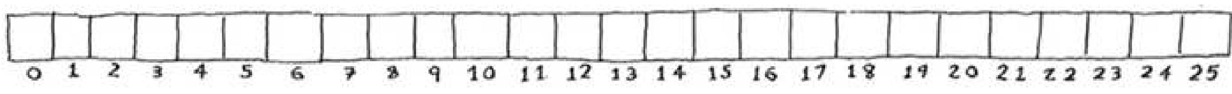
- [illegible]
- [illegible]
- [illegible]/[illegible]

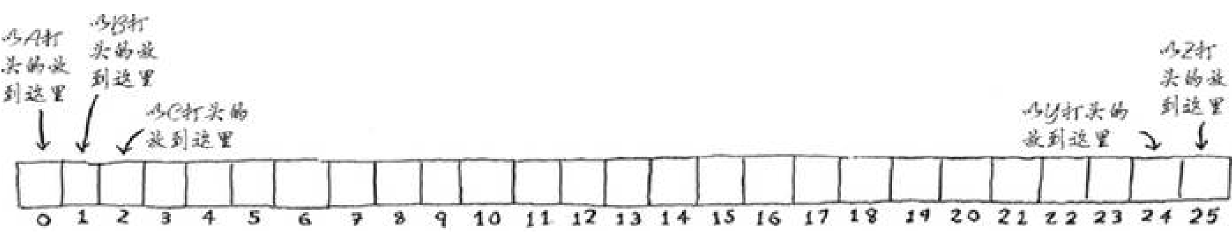
5.3 [illegible]

[illegible]

[illegible]

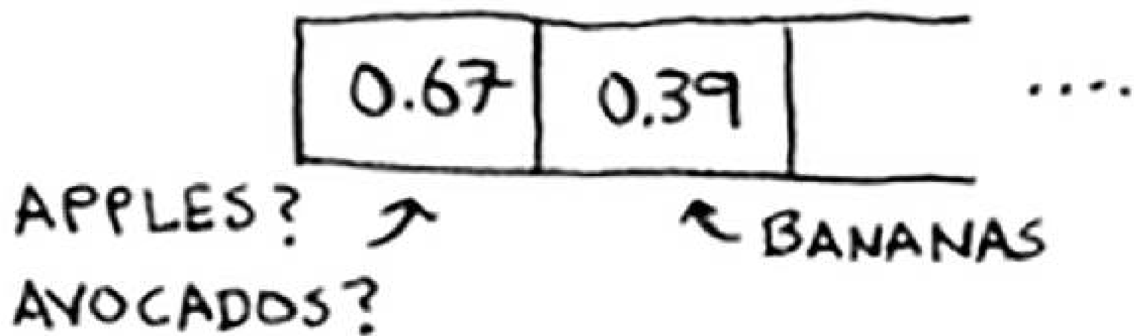



0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25

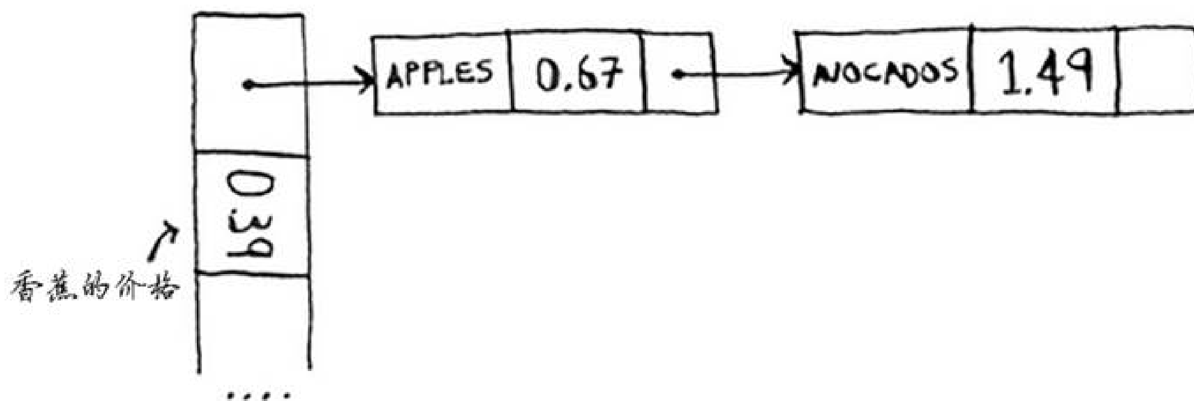

以A打头的放到这里
以B打头的放到这里
以C打头的放到这里
以Y打头的放到这里
以Z打头的放到这里
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25


0.67
APPLES

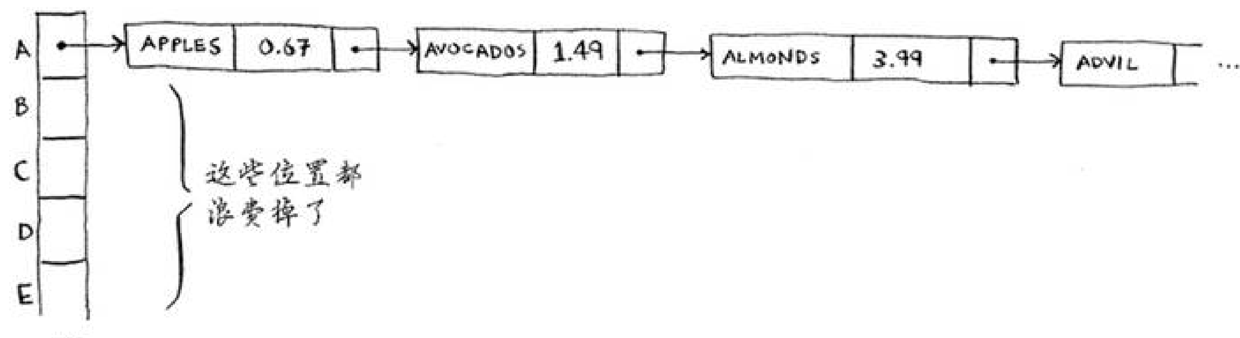

0.67
0.39
APPLES
BANANAS



[illegible]（collision）[illegible]



[illegible]apple[illegible]avocado[illegible]apple[illegible]——[illegible]A[illegible]



[illegible]

[illegible]

- [illegible]
- [illegible]

[illegible]

5.4 [illegible]

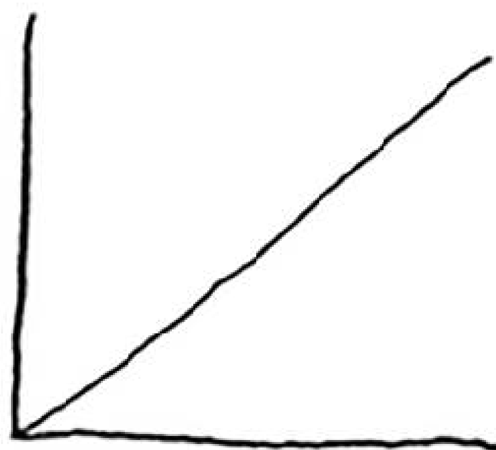
[illegible]

	平均情况	最糟情况
查找	O(1)	O(n)
插入	O(1)	O(n)
删除	O(1)	O(n)

散列表的性能

[illegible] $O(1)$ [illegible] $O(1)$ [illegible]

[illegible]



O(n)
线性时间
（简单查找）

[illegible]



O(log n)
对数时间
（二分查找）

[illegible]



O(1)

常量时间

（散列表）

[illegible]10[illegible]——[illegible]

[illegible]$O(n)$——[illegible]

	散列表（平均情况）	散列表（最糟情况）	数组	链表
查找	O(1)	O(n)	O(1)	O(n)
插入	O(1)	O(n)	O(n)	O(1)
删除	O(1)	O(n)	O(n)	O(1)

[illegible]

- [illegible]
- [illegible]

[illegible]

[illegible]

5.4.1 [illegible]

[illegible]

$$\frac{\text{散列表包含的元素数}}{\text{位置总数}}$$

[illegible]2/5[illegible]0.4[illegible]

已占用的位置

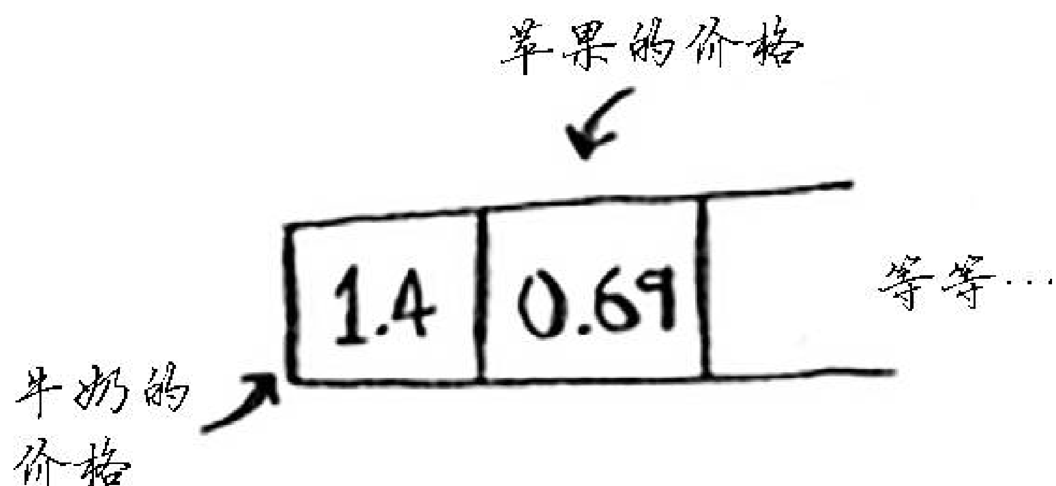
填装因子 = 2/5

[illegible]

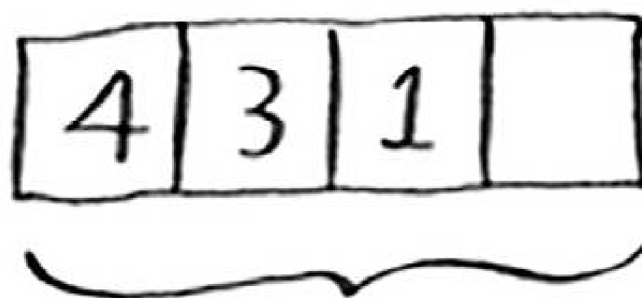
填装因子

[illegible]1/3[illegible]

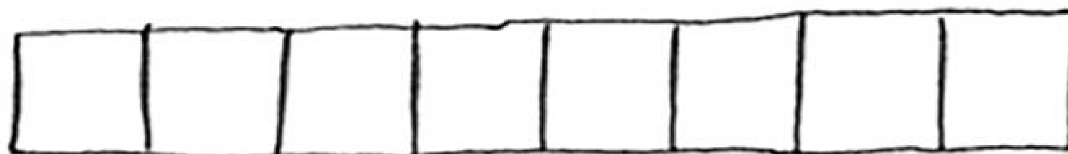
[illegible]100[illegible]100[illegible]



[illegible]1[illegible]50[illegible]2[illegible]
[illegible]1[illegible]
[illegible]（resizing[illegible]



[illegible]



[illegible]hash [illegible]

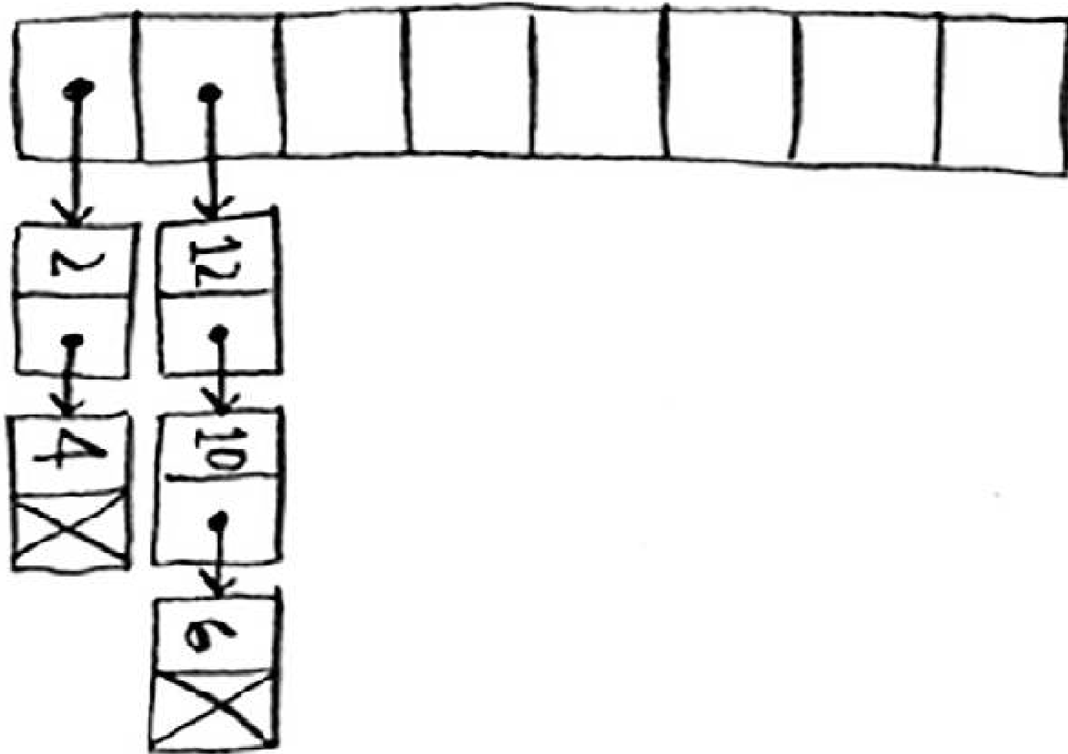
[illegible]3/8[illegible]
[illegible]0.7[illegible]

[illegible]
[illegible]$O(1)$[illegible]

5.4.2 [illegible]

[illegible]

[illegible]



[illegible]——[illegible]SHA[illegible]
[illegible]

[illegible]

[illegible]

[illegible]

A. [illegible]1[illegible]

B. [illegible]

C. [illegible]a[illegible]

D. [illegible]a = 2[illegible]b = 3[illegible]c = 5[illegible]d = 7[illegible]e = 11[illegible]
[illegible]10[illegible]bag [illegible](3 + 2 + 17) % 10 = 22 % 10 = 2[illegible]

□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□10□

5.5 □□□□□□□□□□□□□□□□□□□□□□□□□□□□Esther□Ben□Bob□Dan□

5.6 □□□□□□□□□□□□□□□□□□A□AA□AAA□AAAA□

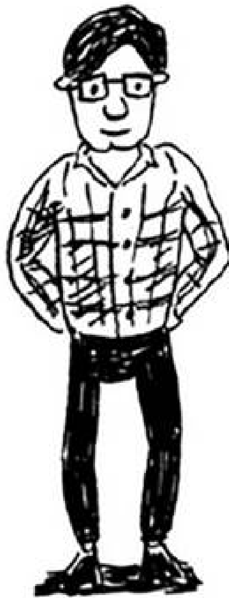
5.7 □□□□□□□□□□□□□□□□*Maus* □*Fun Home* □*Watchmen* □

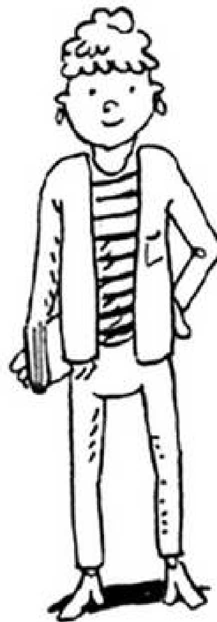
5.5 □□

□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□Python□□□□□□□□□□□□□□□□□□□□□□□□□□□□

□□□

- □□□□□□□□□□□□□□□□□□□
- □□□□□□□□□□□□□□□□□□□□□□□□□□
- □□□□□□□□□□□□□□□□□□□
- □□□□□□□□□□□□□□
- □□□□□□□□0.7□□□□□□□□□□□□
- □□□□□□□□□□□□□□□Web□□□□□□
- □□□□□□□□□□□□□□









[illegible] 6 [illegible] [illegible]

[illegible]

- [illegible]
- [illegible]“[illegible]X[illegible]”[illegible]
- [illegible]
- [illegible]

[illegible]*X*[illegible]*Y*[illegible]——[illegible]
[illegible]breadth-first search[illegible]BFS[illegible]

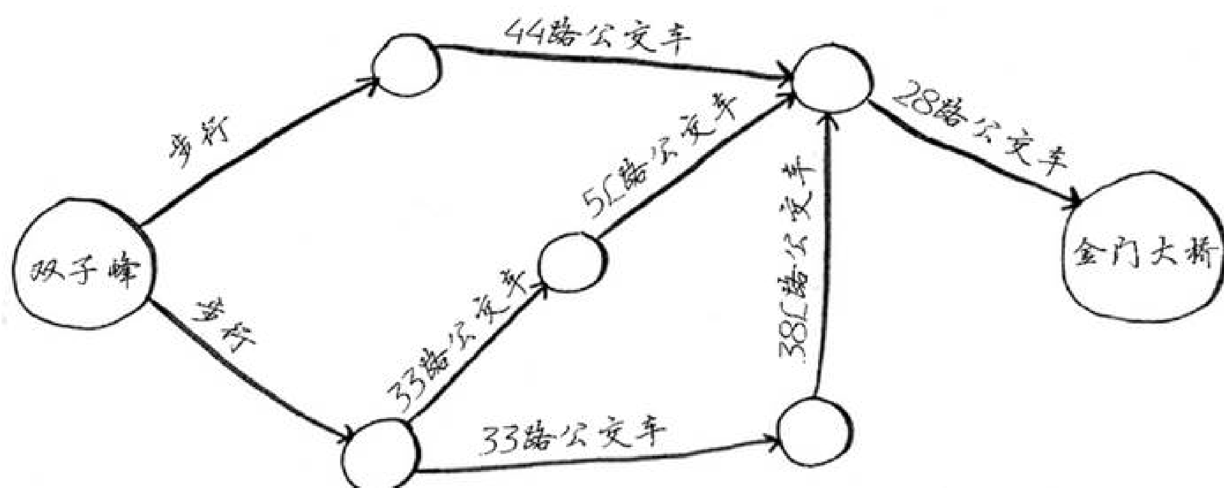
[illegible]

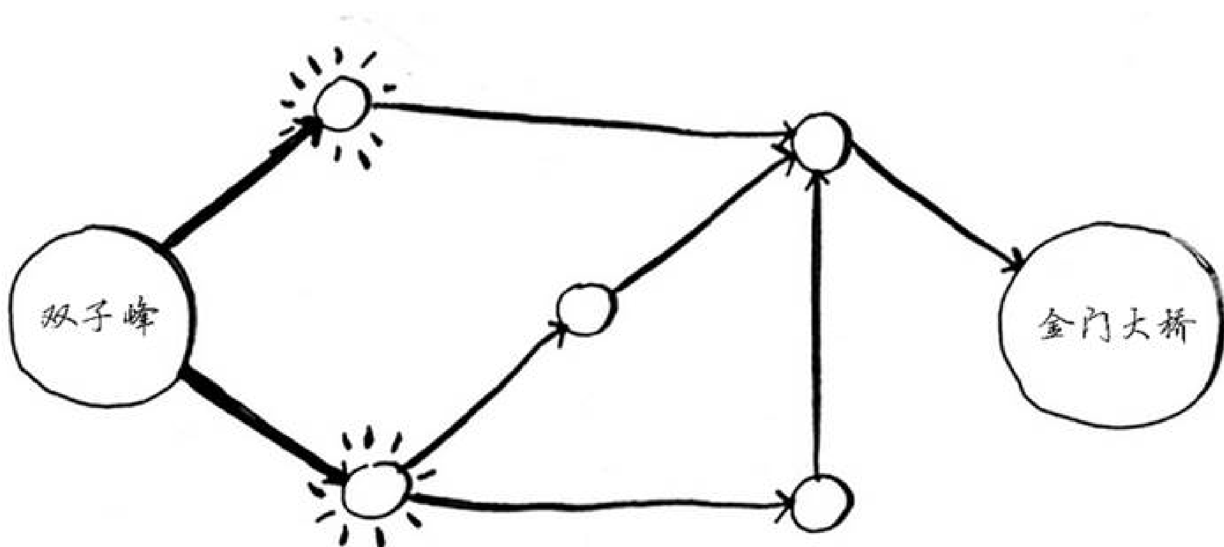
- [illegible]AI[illegible]
- [illegible]READED[illegible]READER[illegible]
- [illegible]

[illegible]

6.1 [illegible]

[illegible]


44路公交车
步行
5L路公交车
28路公交车
双子峰
38L路公交车
金门大桥
步行
33路公交车
33路公交车


双子峰
金门大桥

双子峰

金门大桥

[illegible]

双子峰

金门大桥

[illegible]

第一步

第二步

第三步

步行

乘44路公交车

乘28路公交车

双子峰

金门大桥

[illegible] （shorterst-path problem）[illegible]

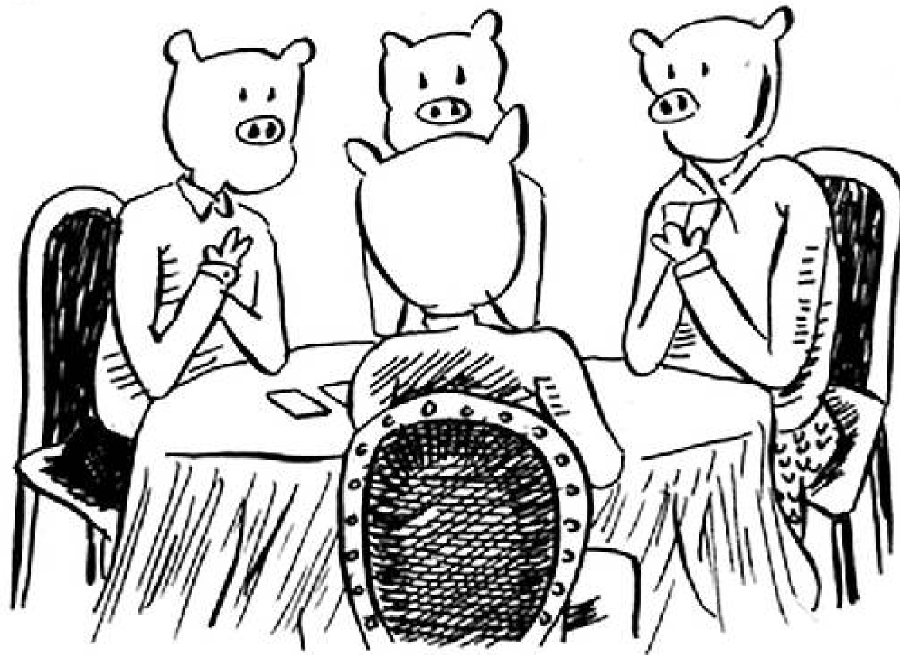
[illegible]

(1) [illegible]

(2) [illegible]

[illegible]

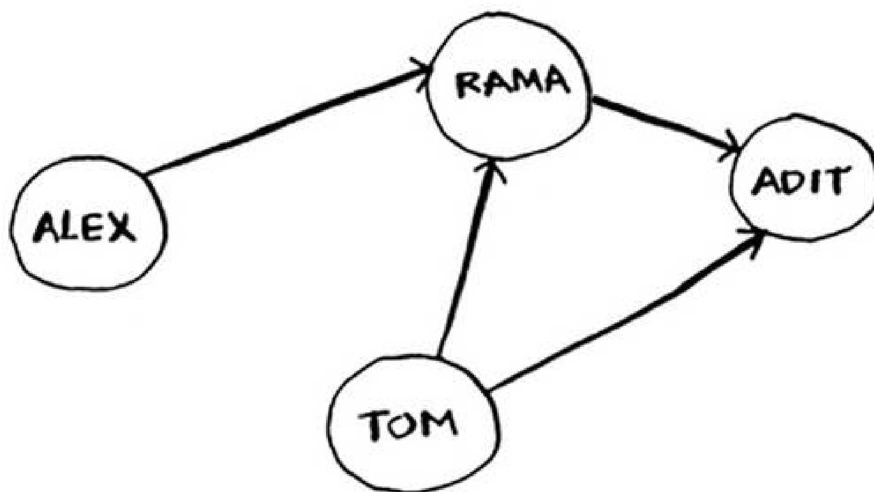
6.2 [illegible]



[illegible]Alex[illegible]Rama[illegible]

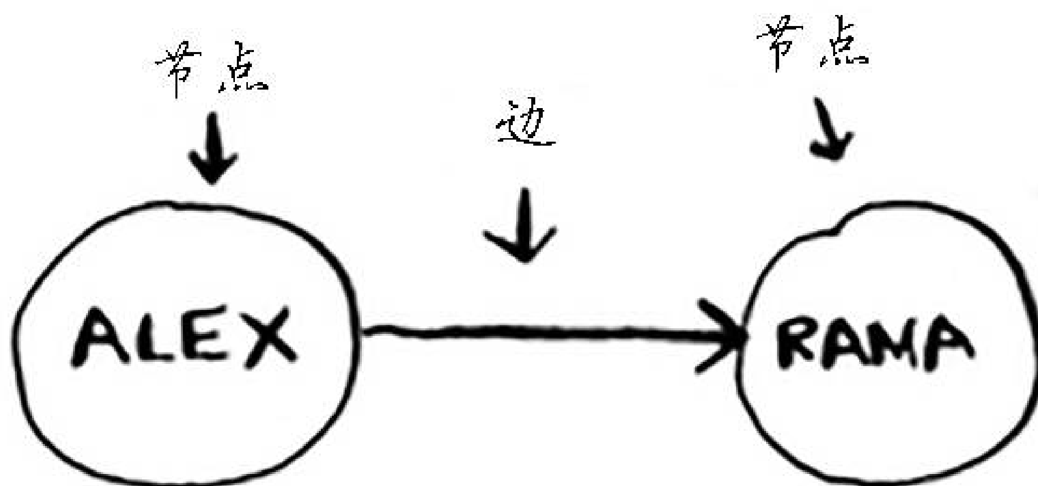


□□□□□□□□□□□□□□□□□□



指出谁欠谁钱的图

Alex□Rama□□Tom□Adit□□□□□□□□□ □node□□□ □edge□□□□



□□ □□□□□□□□□□
Rama□Alex□□□□Adit□□Alex□□□□□□□□□□□□□□□Adit□□Rama□□□□
□□Tom□□□□

□□□□□□□□□□□□□□□□□□□□□□□□□□□□□

6.3 □□□□□□

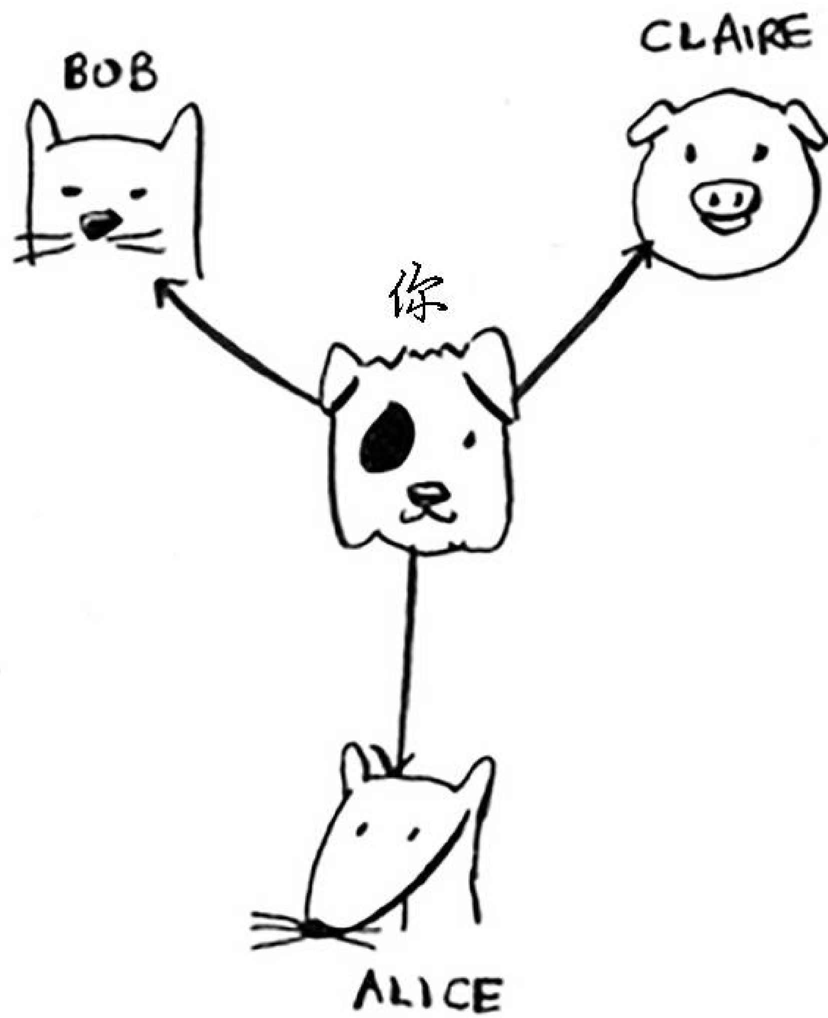
□1□□□□□□□□□□——□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□

- □□□□□□□□□A□□□□□□□□B□□□□□
- □□□□□□□□□A□□□□□□□B□□□□□□□□

□□□
□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□



□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□Facebook□□□□□□□□□□
□□□□□□□□□□□□□□□


BOB
CLAIRE
你
ALICE


ALICE
BOB
CLAIRE

ALICE
BOB
CLAIRE
ALICE是芒果销售商吗?
是：大功告成
否
BOB
CLAIRE
BOB是芒果销售商吗?
是：大功告成
否
CLAIRE
CLAIRE是芒果销售商吗?
是：大功告成
否：没有朋友是芒果销售商

[illegible]

[illegible]Alice[illegible]

6.3.1 [illegible]

□□□□□□□□□□□□□□□□□□□□□

- □□□□□□□□□□A□□□□□□□□□B□□□□□□□□□□□□□□□□□□□□□□□□□□□
- □□□□□□□□□□A□□□□□□□□B□□□□□□□□□□□□□□□□□□□□□□□□□□□

□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□——□□□□□□□□□□□□□□□□□□□□□
□□□□□□□□□□□□□□□□

□□
□□
□□□Claire□□Anuj□□
Claire□□□□□□□□Anuj□□□□□□□□□□□□□Claire□□□□Anuj□

□□□□□□□□□□□□□□□□□□□□□□□□□□□□

□□
□□□□□□□□□□□□□□□□□□□□□□□□□□□A□B□□□□□□□□□□□□□□□□



□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□Claire□□Anuj□□□□□□□□□□
□□Claire□□□□Anuj□□Claire□Anuj□□□□□□□□□□□□□Anuj□□□□
Claire□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□Anuj□□□□□□□□□Claire
□□
□queue□□

6.3.2 □□

[illegible]

[illegible]

[illegible] First In First Out [illegible] FIFO [illegible] Last In First Out [illegible] LIFO [illegible]

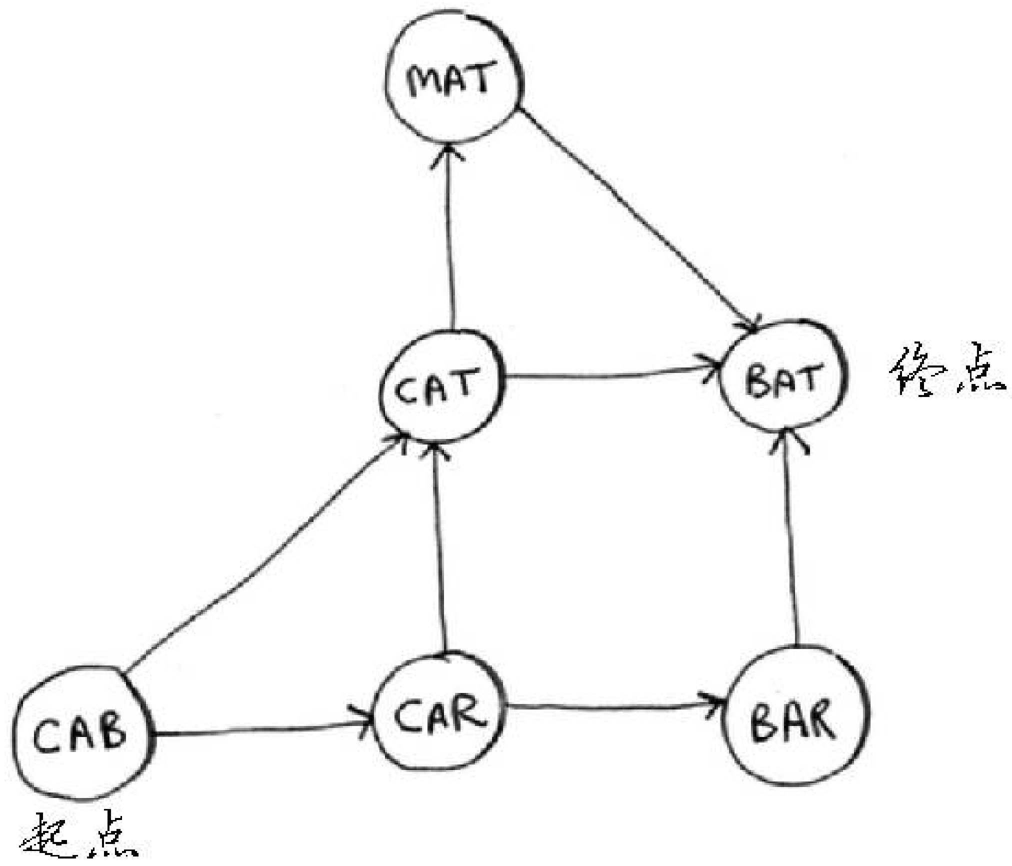
[illegible]

[illegible]

[illegible]

6.1 [illegible]

6.2 [illegible]cab[illegible]bat[illegible]



6.4 [illegible]

[illegible]

[illegible]“[illegible]→Bob”[illegible]

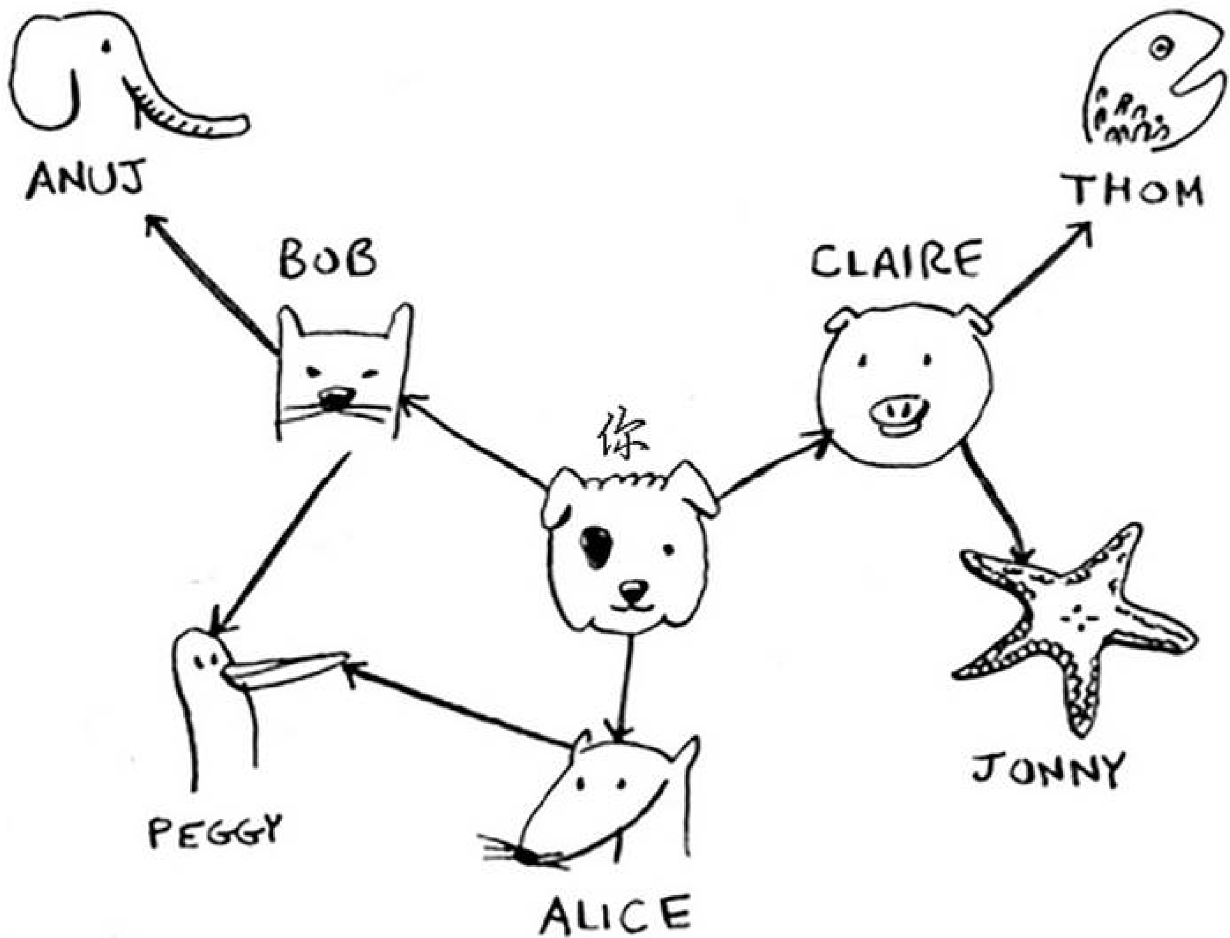
[illegible]

[illegible]Python[illegible]

```
graph = {}
graph["you"] = ["alice", "bob", "claire"]
```

□□□“□”□□□□□□□□□□□□□graph["you"] □□□□□□□□□□□□“□”□□□□□□

□□□□□□□□□□□□□□□□□□Python□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□



□□□□Python□□□□□

```
graph = {}
graph["you"] = ["alice", "bob", "claire"]
graph["bob"] = ["anuj", "peggy"]
graph["alice"] = ["peggy"]
graph["claire"] = ["thom", "jonny"]
```

```
graph["anuj"] = []
graph["peggy"] = []
graph["thom"] = []
graph["jonny"] = []
```

[illegible]

```
graph["claire"] = ["thom", "jonny"]
graph["anuj"] = []
```

[illegible]

```
graph["anuj"] = []
graph["claire"] = ["thom", "jonny"]
```

[illegible]

[illegible]

Anuj [illegible] Peggy [illegible] Thom [illegible] Jonny [illegible] (directed graph) [illegible] Anuj [illegible] Bob [illegible] Bob [illegible] Anuj [illegible] (undirected graph) [illegible]

ROSS RACHEL

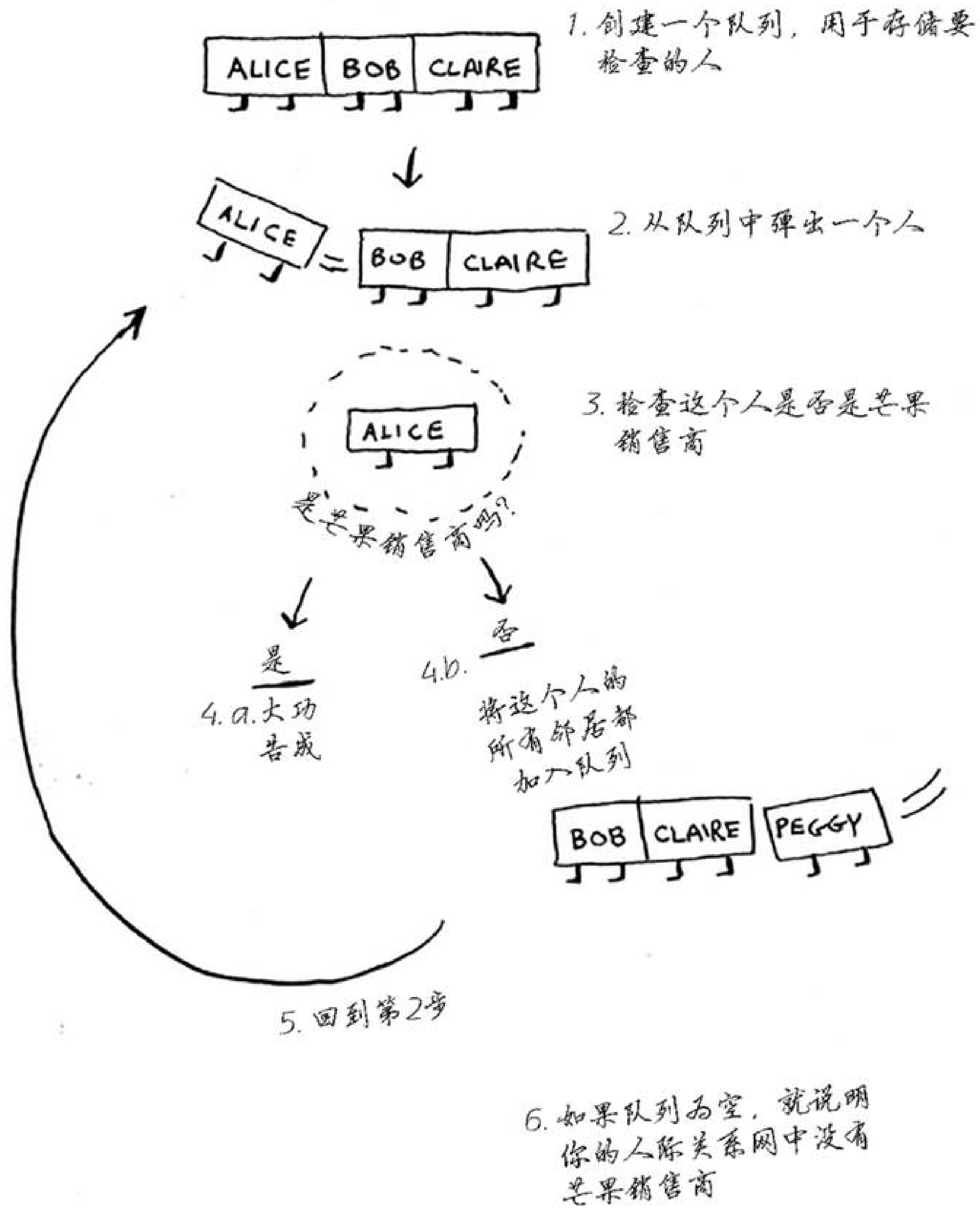
有向图

ROSS RACHEL

无向图

6.5 [illegible]

□□□□□□□□□□□□□□□□□



1. 创建一个队列，用于存储要检查的人
ALICE
BOB
CLAIRE
2. 从队列中弹出一个人
3. 检查这个人是否是芒果销售商
是芒果销售商吗?
是
4.a. 大功告成
否
4.b. 将这个人的所有邻居都加入队列
PEGGY
5. 回到第2步
6. 如果队列为空，就说明你的人际关系网中没有芒果销售商

□□

□□□□□□□□□□□“□□”□“□□”□□□□□□□□□“□□”□“□□”□□□□□□□□□
□□□□□□□□□□□□□

□□□□□□□□□□□Python□□□□□□□deque □□□□□□□□□□

```
from collections import deque
search_queue = deque()  ←------------□□□□□□
search_queue += graph["you"]  ←------□□□□□□□□□□□□□□□□
```

□□□□graph["you"] □□□□□□□□□□□□□□□□□["alice", "bob", "claire"] □□□□□□□□□□□□□□□



□□□□□□□□□□□

```
while search_queue:  ←------□□□□□□□
    person = search_queue.popleft()  ←------□□□□□□□□□□
    if person_is_seller(person):  ←------□□□□□□□□□□□□□
        print person + " is a mango seller!"  ←-------□□□□□□
        return True
    else:
        search_queue += graph[person]  ←------□□□□□□□□□□□□□□□□□□□□□
□
return False  ←------□□□□□□□□□□□□□□□□□□□□□□
```

`person_is_seller`

```
def person_is_seller(name):
    return name[-1] == 'm'
```

m

当前的 search_queue

```
search_queue = deque()
search_queue += graph["you"]
while search_queue:
person = search_queue.popleft()
if person_is_seller(person):
else:
search_queue += graph[person]
while search_queue:
person = search_queue.popleft()
if person_is_seller(person):
else:
search_queue += graph[person]
```

空的

ALICE CLAIRE BOB

只要搜索队列不是空的

当前的人为Alice

Alice不以m结尾，因此不是芒果销售商

CLAIRE BOB PEGGY

BOB PEGGY THOM JONNY

…等等…

[illegible]

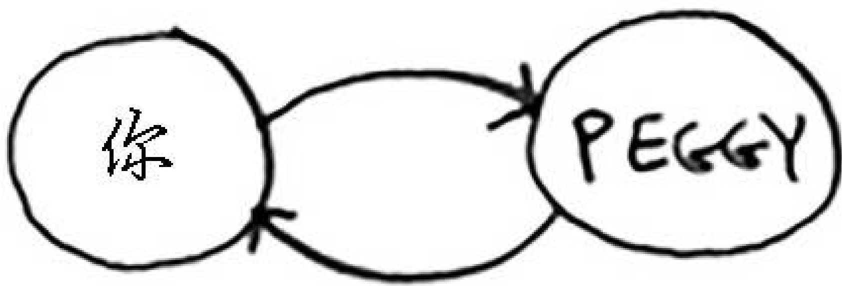
- [illegible]
- [illegible]

Peggy□□Alice□□□□□Bob□□□□□□□□□□□□□□□□□□□□□□Alice□□□□□□□□□□□□Bob□□□□□□□□□□□□□□□□□Peggy□



□□□□□□Peggy□□□

□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□



□□□□□□□□□□□□□□□□□



[illegible]Peggy[illegible]



[illegible]



[illegible]Peggy[illegible]





```
def search(name):
    search_queue = deque()
    search_queue += graph[name]
    searched = []  ←------------------------------
    while search_queue:
```

```
        person = search_queue.popleft()
        if person not in searched:     ←----------[illegible]
            if person_is_seller(person):
                print person + " is a mango seller!"
                return True
            else:
                search_queue += graph[person]
                searched.append(person)    ←------[illegible]
    return False


search("you")
```

[illegible]person_is_seller [illegible]

[illegible]

[illegible] O ([illegible])[illegible]

[illegible] $O(1)$ [illegible] O ([illegible]) [illegible] O ([illegible] + [illegible]) [illegible] $O(V+E)$ [illegible] V [illegible]vertice[illegible] E [illegible]

[illegible]

[illegible]

[illegible]“[illegible]”[illegible]“[illegible]”[illegible]

[illegible]

(1) [illegible]

(2) [illegible]

(3) [illegible]

(4) [illegible]

[illegible]“[illegible]”[illegible]

(1) [illegible]

(2) [illegible]

(3) [illegible]

(4) [illegible]

6.3 [illegible]

A.

1. 起床
2. 洗澡
3. 吃早餐
4. 刷牙

B.

1. 起床
2. 刷牙
3. 吃早餐
4. 洗澡

C.

1. 洗澡
2. 起床
3. 刷牙
4. 吃早餐

6.4 [illegible]

[illegible]A[illegible]B[illegible]A[illegible]B[illegible]

[illegible]

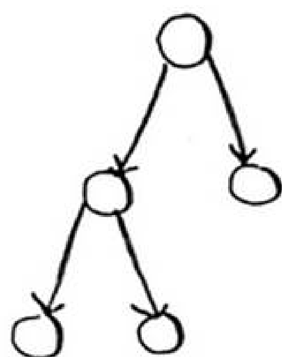
你

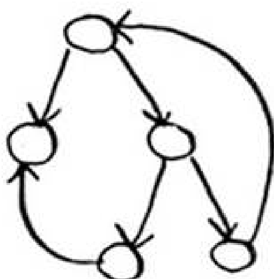
父母

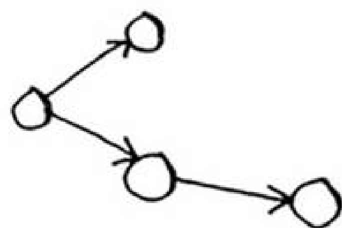
祖父母



6.5 □□□□□□□□□□□□







6.6 □□

- □□□□□□□□□□□□□A□B□□□□

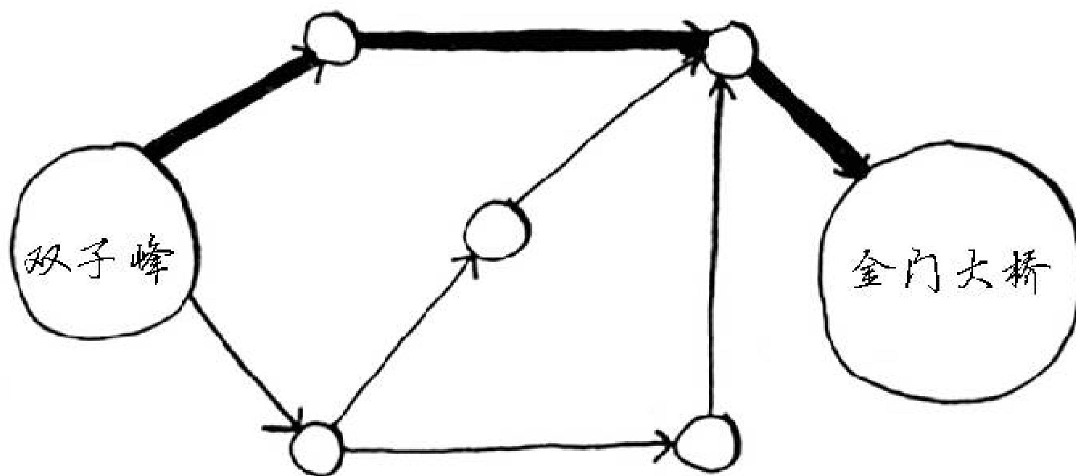
- [illegible]
- [illegible]
- [illegible]rama→adit[illegible]rama[illegible]adit[illegible]
- [illegible]ross - rachel[illegible]“ross[illegible]rachel[illegible]rachel[illegible]ross[illegible]”[illegible]
- [illegible]FIFO[illegible]
- [illegible]LIFO[illegible]
- [illegible]
- [illegible]

第 7 章　狄克斯特拉算法

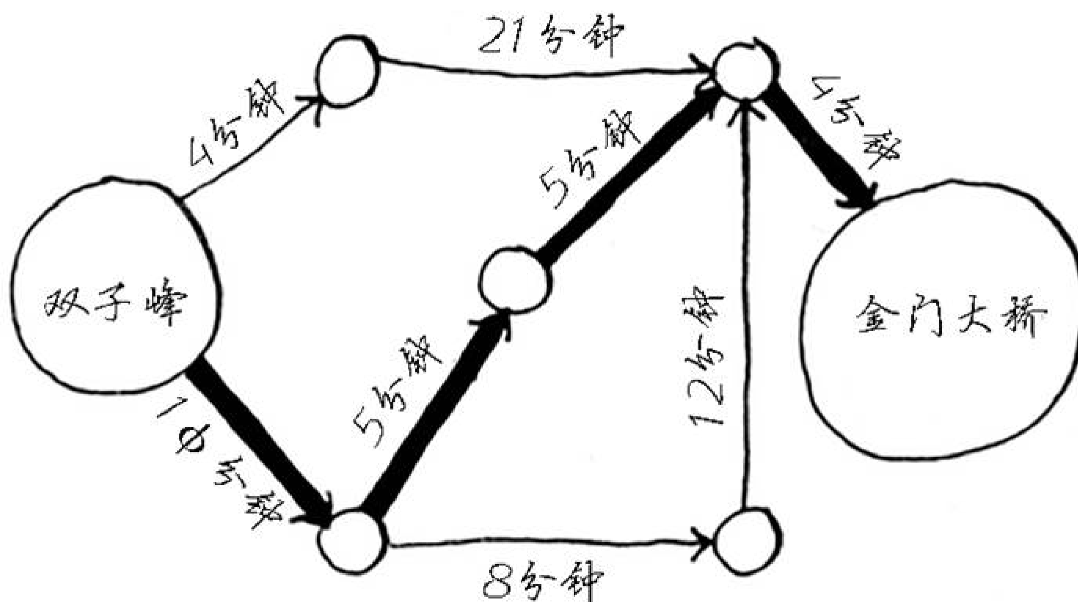
本章内容

- 继续图的讨论，介绍加权图——提高或降低某些边的权重。
- 介绍狄克斯特拉算法，让你能够找出加权图中前往X的最短路径。
- 介绍图中的环，它导致狄克斯特拉算法不管用。

在前一章，你找出了从A点到B点的路径。



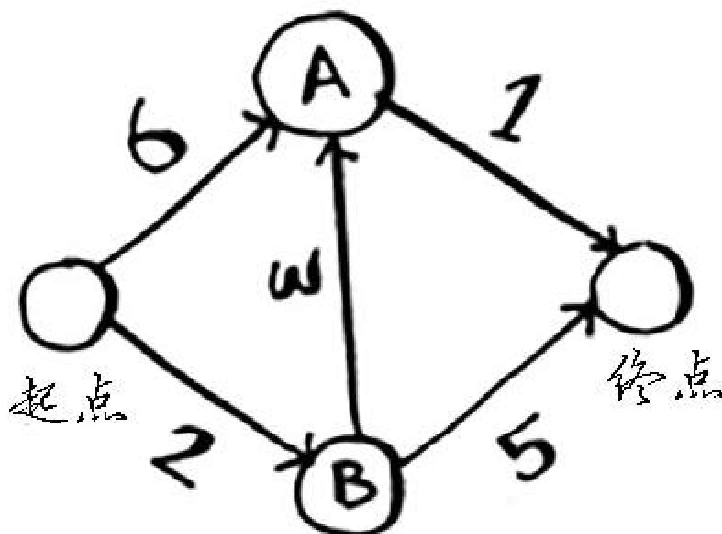
这是最短路径，因为段数最少——只有三段，但不一定是最快路径。如果给这些路段加上时间，你将发现有更快的路径。



[illegible] ——[illegible]（Dijkstra's algorithm）[illegible]

7.1 [illegible]

[illegible]



[illegible]

[illegible]

[illegible]7[illegible]4[illegible]

(1) [illegible]“[illegible]”[illegible]

(2) [illegible]

(3) [illegible]

(4) [illegible]

[illegible] [illegible]A[illegible]B[illegible]

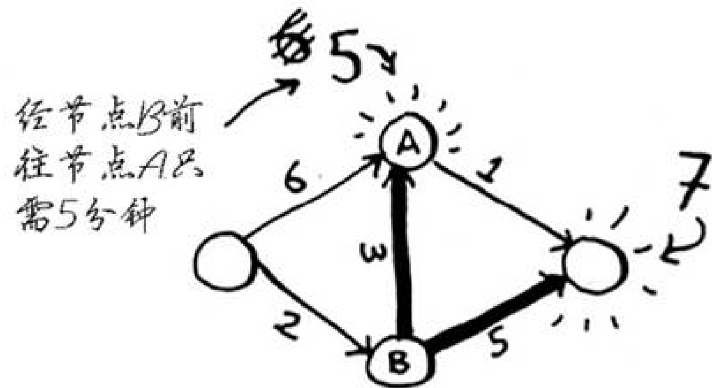
□□□□A□□6□□□□□□□□B□□2□□□□□□□□□□□□□□□□□□□□□□□□

节点	耗时
A	6
B	2
终点	∞

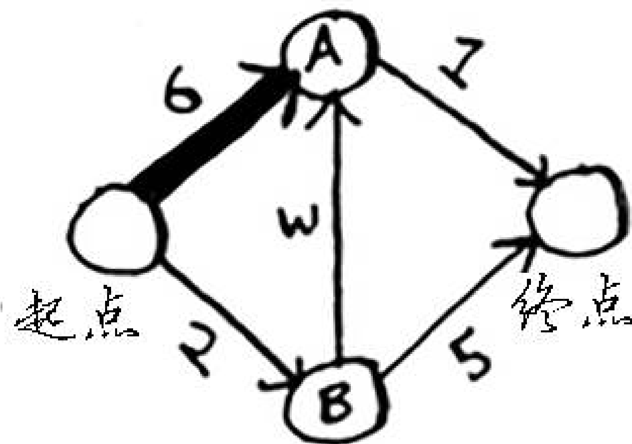
□□□B□□□□
——2□□□□□□□

□□□ □□□□□□B□□□□□□□□□□□□□

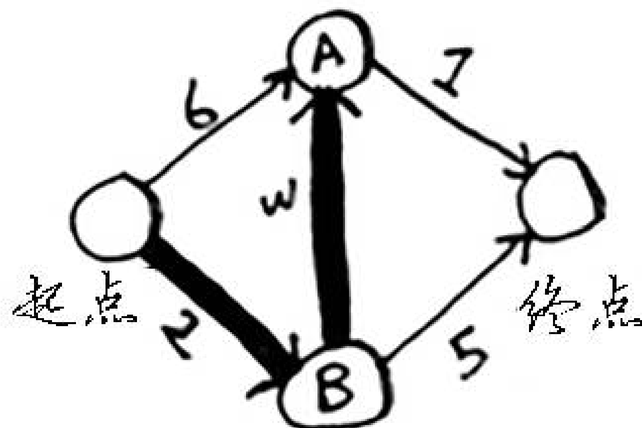
节点	耗时
A	~~6~~ 5
B	2
终点	7



□□□□□□□□□□□□A□□□□□□□□□□□□□A□□6□□□



□□□□□B□□□□A□□5□□□



□□□□B□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□

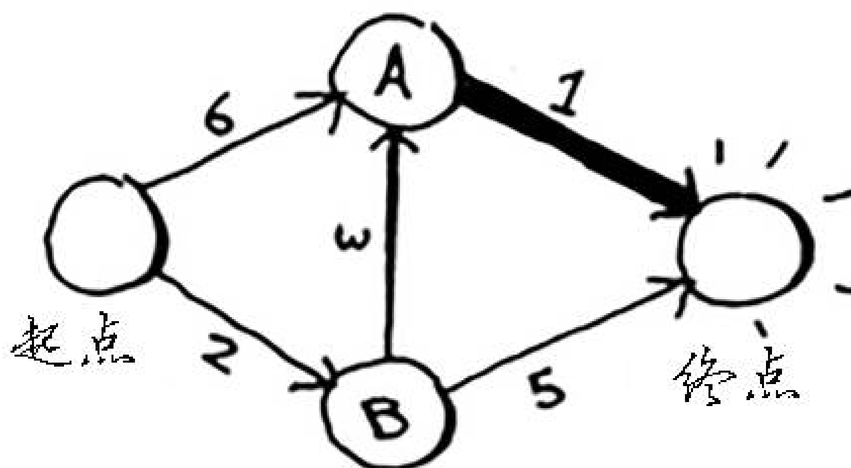
- □□□□A□□□□□□□□□6□□□□□5□□□□
- □□□□□□□□□□□□□□□□□□□7□□□□

□□□ □□□□

□□□□□ □□□□□□□□□□□□□□□□□□□□B□□□□□□□□□□B□□□□□□□□□□□□□ □□□□A□

节点	耗时
A	5
B	2
终点	7

□□□□□ □□□□□A□□□□□□□□□



□□□□□□□□□□□6□□□

[illegible]

- [illegible]B[illegible]2[illegible]
- [illegible]A[illegible]5[illegible]
- [illegible]6[illegible]

节点	耗时
A	5
B	2
终点	6

[illegible]——[illegible]

[illegible]3[illegible]

广度优先搜索找出的最短路径

□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□“□□□□”□□

加权图（使用狄克斯特拉算法）

非加权图（使用广度优先搜索）

□□□□□□□□□□□□□□□□□□4□□□□

(1) □□□□□□□□□□□□□□□□□□□□□□□□

(2) □□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□

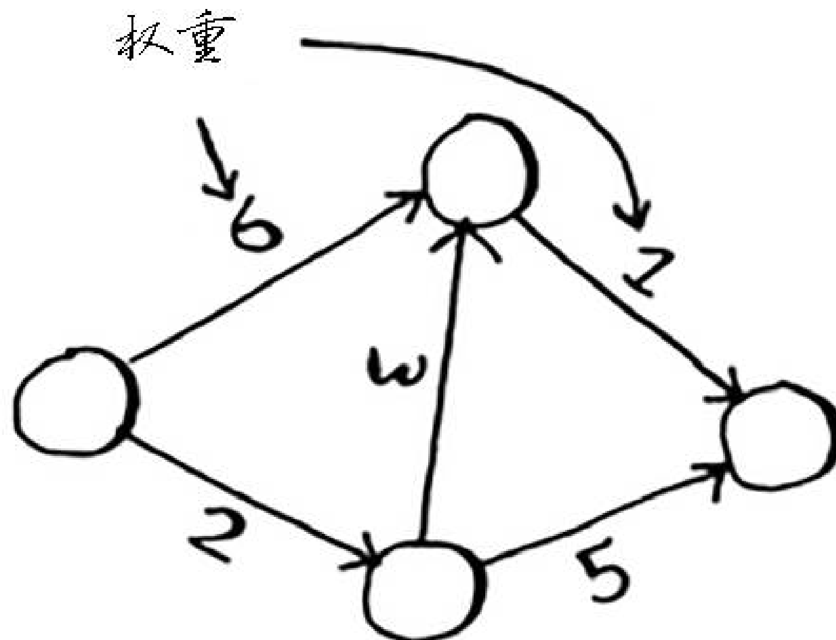
(3) □□□□□□□□□□□□□□□□□□□□□□□□

(4) □□□□□□□□□□□□□□□□□

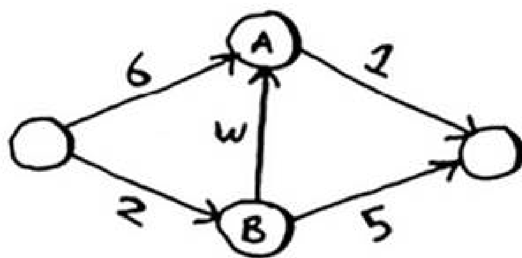
7.2 [illegible]

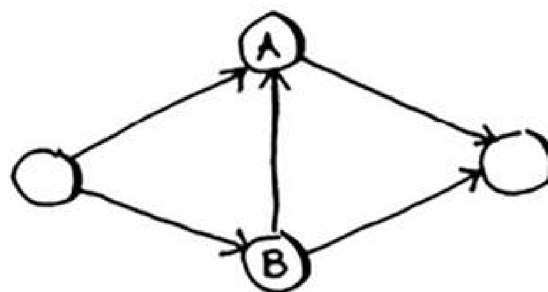
[illegible]

[illegible] [illegible]weight[illegible]



[illegible] [illegible]weighted graph[illegible] [illegible]unweighted graph[illegible]





[illegible] [illegible] [illegible] [illegible]

环！
你可从A点出发，走一圈后又回到A点
A
B
C

A
环！
4
4
B
2
3
起点
终点

总权重:
5

总权重:
13

□□□□□□□□□□□□□8□□□□□□□□□□□□□□□□□□□

□□□□□□□6□□□□□□□□□□□□□□

□□□□□□□□□□□□□□□□□□□□□□□

□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□ □directed acyclic graph□DAG□□

7.3 □□□

□□□□□□□□□□□□□□□□□□□□□□Rama□□□□□□□□□□□□



Alex□□“□□□□□□□□□Destroyer□□□□□□□□□□□□□□□□□□□□5□□□□□□□□□□□□□□□Rick Astley□□□□□”Amy□□“□□□”



Beethoven□□□“□□□□□□□□□□□□□□□□□Amy□□□□□□□□”

[illegible]Rama[illegible]

[illegible]30[illegible]15[illegible]Rama[illegible]

[illegible]

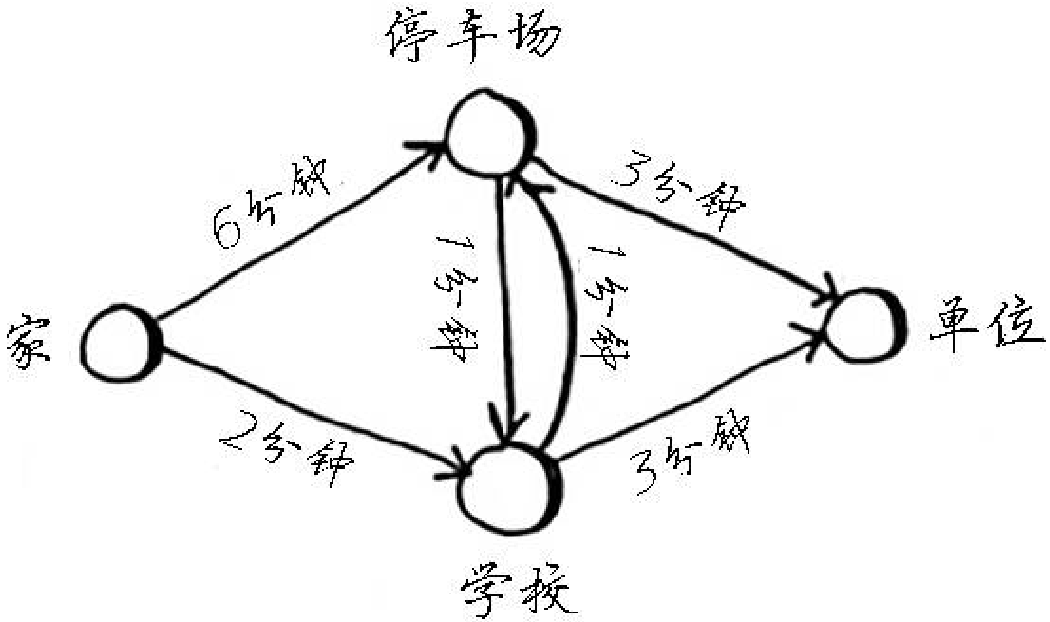
节点	开销
黑胶唱片	5
海报	0
吉他	∞
架子鼓	∞
钢琴	∞

我们还不知道如何从起点前往这些节点

[illegible]

节点	父节点
黑胶唱片	乐谱
海报	乐谱
吉他	—
架子鼓	—
钢琴	—

[illegible]



停车场
家
单位
学校
6分钟
3分钟
1分钟
1分钟
2分钟
3分钟

父节点	节点	开销
乐谱	黑胶唱片	5
乐谱	海报	0
海报	吉他	30
海报	架子鼓	35
—	钢琴	∞

我们经“海报”前往这些节点

父节点	节点	开销
乐谱	黑胶唱片	5
乐谱	海报	0
海报	吉他	30
海报	架子鼓	35
—	钢琴	∞

[illegible]——[illegible]5[illegible]

[illegible]

父节点	节点	开销
乐谱	黑胶唱片	5
乐谱	海报	0
黑胶唱片	吉他	~~30~~ 20
黑胶唱片	架子鼓	~~35~~ 25
—	钢琴	∞

[illegible]“[illegible]”[illegible]“[illegible]”[illegible]“[illegible]”[illegible]

[illegible]

父节点	节点	开销
乐谱	黑胶唱片	5
乐谱	海报	0
黑胶唱片	吉他	20
黑胶唱片	架子鼓	25
吉他	钢琴	40

□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□——□□□□□□□□

父节点	节点	开销
乐谱	黑胶唱片	5
乐谱	海报	0
黑胶唱片	吉他	20
黑胶唱片	架子鼓	25
架子鼓	钢琴	35

□□□□□□□Rama□□□□□□□□□□□□□□□□□□□□**Rama**□□□□□**35**□□ □

□□□□□□□□□□□□□□□□□□□□□□□□□35□□□□□□□□□□□□□□□□ □□□□

父节点	节点
乐谱	黑胶唱片
乐谱	海报
黑胶唱片	吉他
黑胶唱片	架子鼓
架子鼓	钢琴

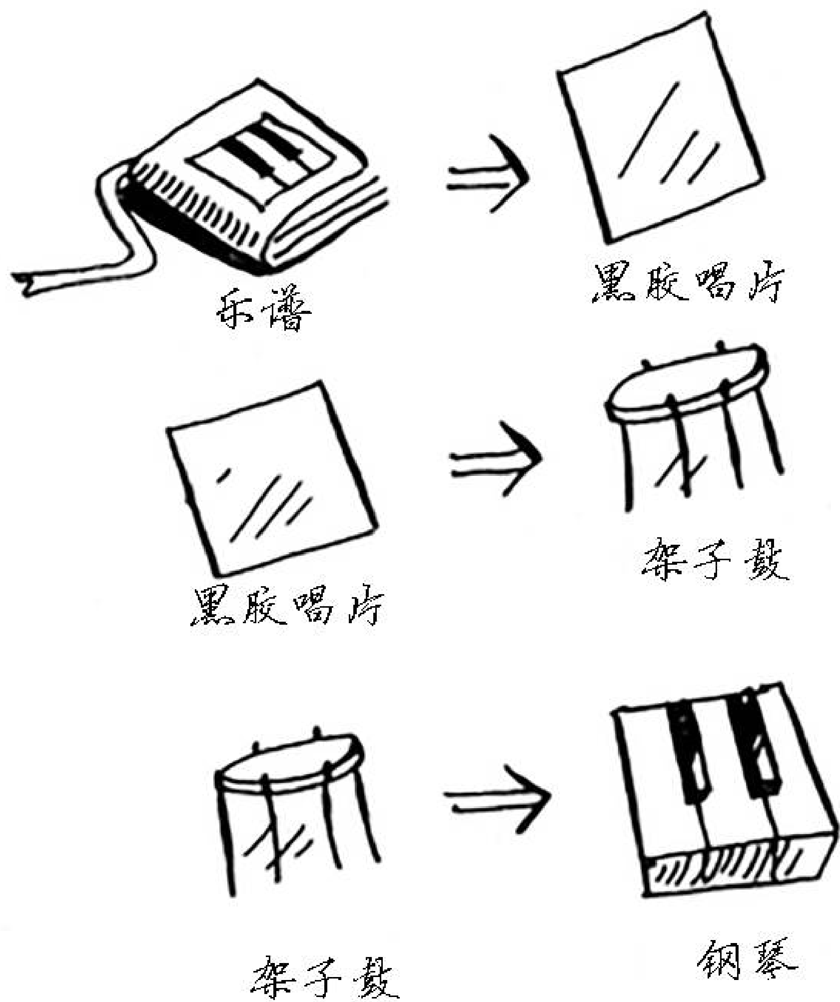
[illegible]Rama[illegible]

[illegible]

[illegible]

□□Rama□□□□□□□□□□□□□□□□□□□□□□
□□□□□□

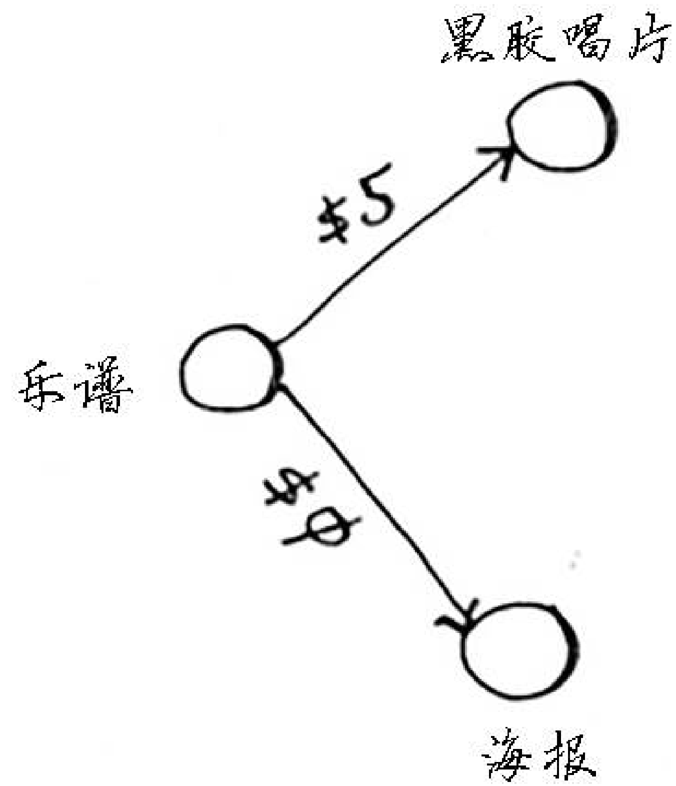
□□□Rama□□□□□□□□□□



[illegible] [illegible]Rama[illegible]

7.4 [illegible]

[illegible]Alex[illegible]



□□□□□□□□Alex□□□□Sarah□□□Sarah□□□□□□□□□7□□□□□□□□□□□□□
Alex□□□□□Rama□□□□Sarah□□□□□□□□□□□□□□□□□□□□□□7□□□□□□
□□□□□□□□□□□□□□□

□□□□□□□□□□□□□□□□□□□□□□Rama□□□□7□□□□□□Rama□□□□□□□□□□□

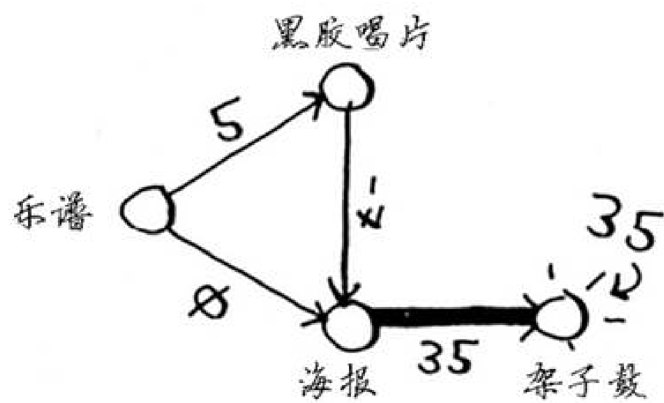
□□□□□□□□——Rama□□2□□□□□□□□□□Rama□□□□□□□□□□□□□□□□□□□□□□□□□

[illegible]2[illegible]Rama[illegible]——[illegible]

黑胶唱片	5
海报	∅
架子鼓	∞

开销

[illegible]



黑胶唱片	5
海报	∅
架子鼓	35

开销

[illegible]35[illegible]

[illegible]

黑胶唱片	5
~~海报~~	~~∅~~
架子鼓	35

[illegible]

开销

黑胶唱片	5
海报	-2
架子鼓	35

[illegible]
[illegible]

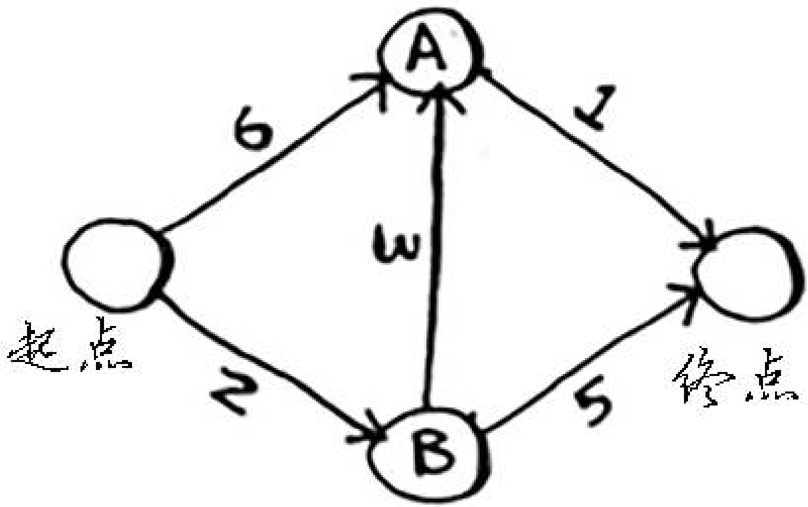
黑胶唱片	5
海报	-2
架子鼓	35

最终开销

[illegible]35[illegible]33[illegible]
[illegible]
[illegible] [illegible]
——[illegible]-[illegible] [illegible]Bellman-Ford algorithm[illegible]
[illegible]

7.5 [illegible]

[illegible]



[illegible]

GRAPH

起点	A	6
	B	2
A	终点	1
B	A	3
	终点	5
终点	—	

COSTS

A	6
B	2
终点	∞

PARENTS

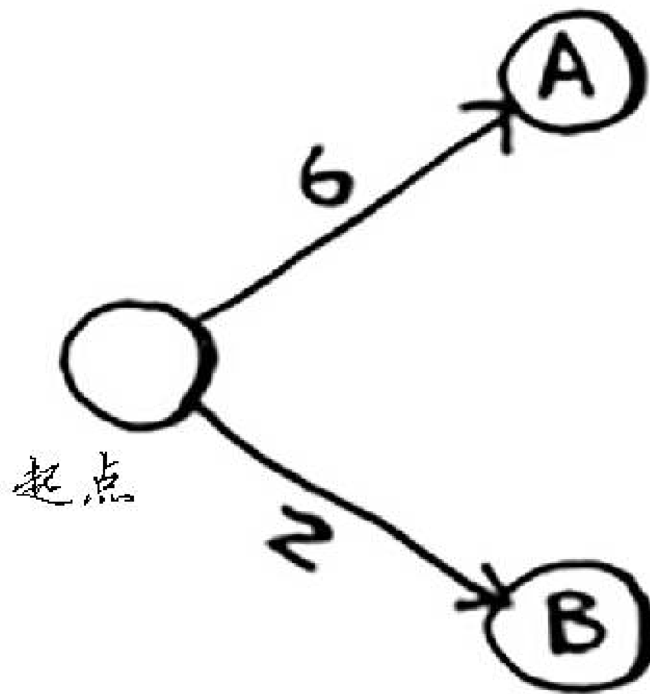
A	起点
B	起点
终点	—

[illegible]costs[illegible]parents[illegible]6[illegible]

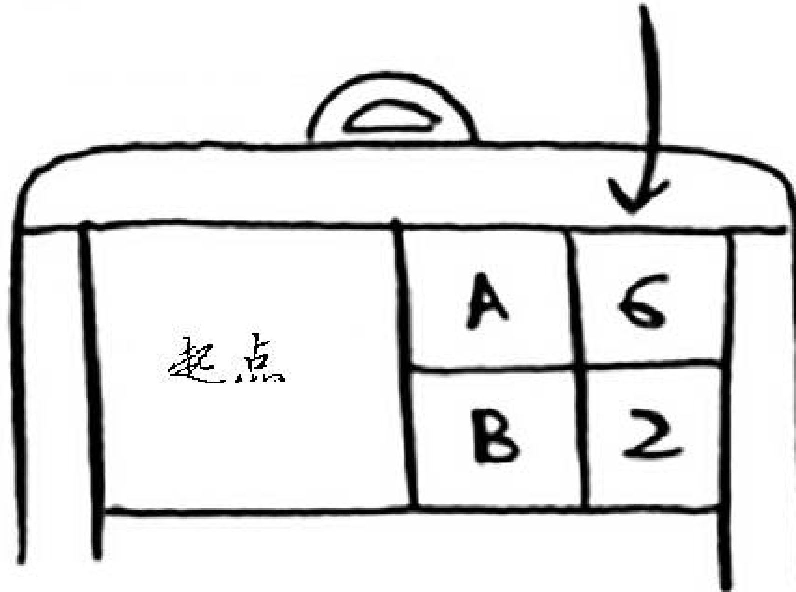
```
graph = {}
```

```
graph["you"] = ["alice", "bob", "claire"]
```

——A B




```
graph["start"] = {}
graph["start"]["a"] = 6
graph["start"]["b"] = 2
```



□□graph["start"] □□□□□□□□□□□□□□□□□□□□□□□□□□□

```
>>> print graph["start"].keys()
["a", "b"]
```

□□□□□□□A□□□□□□□□□□□B□□□□□□□□□□□□□□□□□□□

```
>>> print graph["start"]["a"]
2
>>> print graph["start"]["b"]
6
```

□□□□□□□□□□□□□□

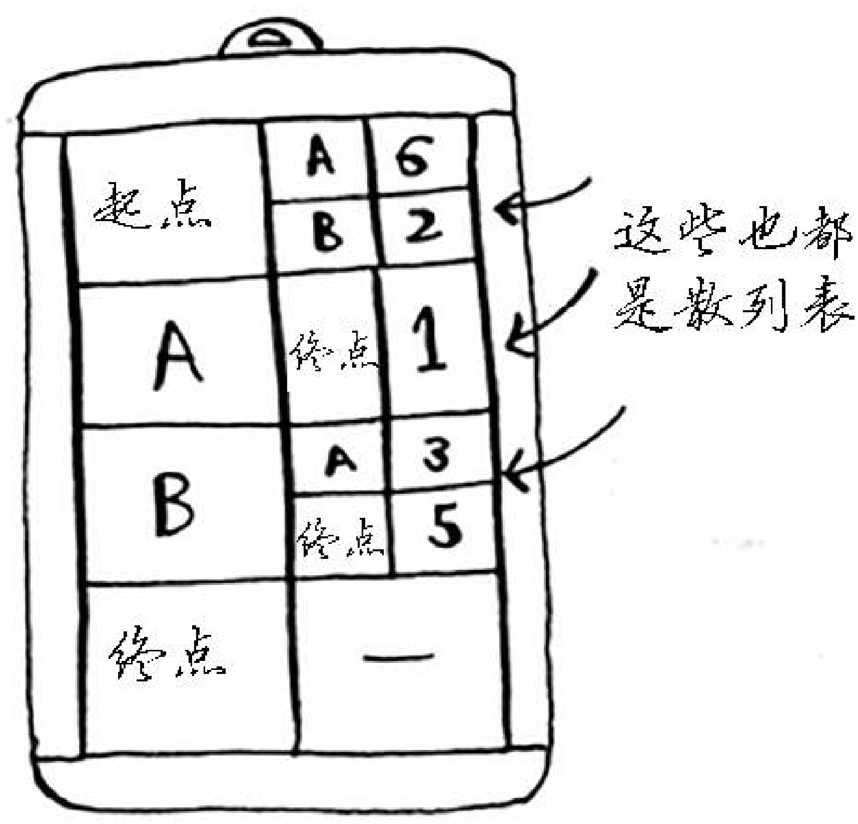
```
graph["a"] = {}
graph["a"]["fin"] = 1

graph["b"] = {}
graph["b"]["a"] = 3
```

```
graph["b"]["fin"] = 5

graph["fin"] = {}  ←-------□□□□□□□□
```

□□□□□□□□□□□□□□□□□



□□□□□□□□□□□□□□□□□□□□□□□

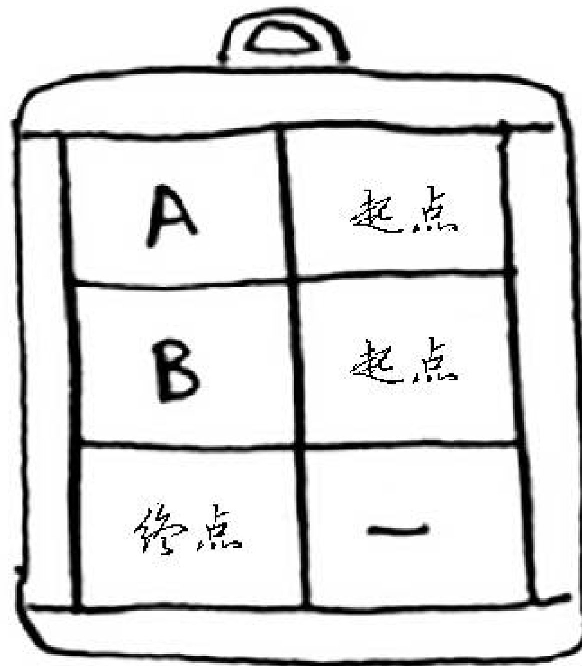
[illegible] B[illegible]2[illegible]A
[illegible]6[illegible]
[illegible]Python[illegible]

```
infinity = float("inf")
```

[illegible]

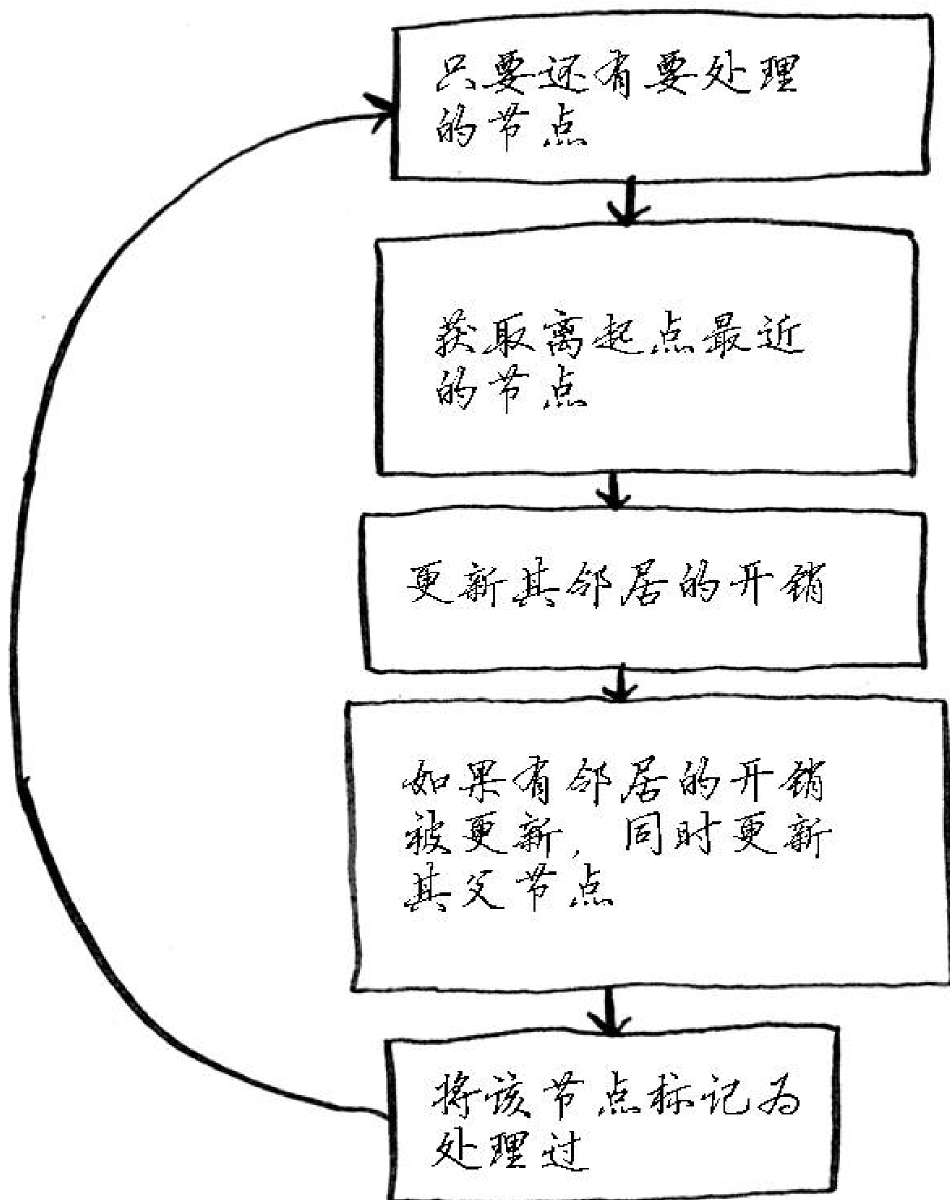
```
infinity = float("inf")
costs = {}
costs["a"] = 6
costs["b"] = 2
costs["fin"] = infinity
```

[illegible]




```
parents = {}
parents["a"] = "start"
parents["b"] = "start"
parents["fin"] = None
```

```
processed = []
```

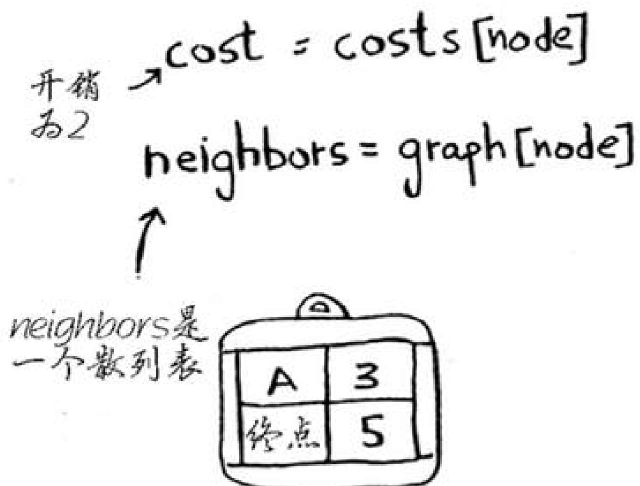


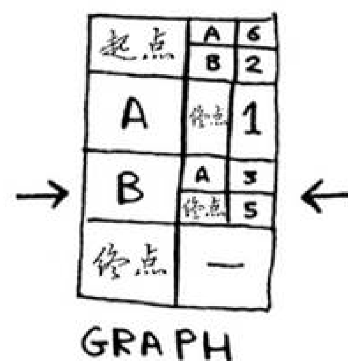

```
node = find_lowest_cost_node(costs)  ←------
while node is not None:  ←------while
    cost = costs[node]
    neighbors = graph[node]
    for n in neighbors.keys():  ←------
        new_cost = cost + neighbors[n]
        if costs[n] > new_cost:  ←------
            costs[n] = new_cost  ←------
            parents[n] = node  ←------
    processed.append(node)  ←------
```

```
node = find_lowest_cost_node(costs)  ←------
```

Python find_lowest_cost_node







```
for n in neighbors.keys():
```

n为"A"

一个节点列表

A	终点

键	值
A	3
终点	5

[illegible]B[illegible]A[illegible]A[illegible]

```
new_cost = cost + neighbors[n]
```

节点B的开销，即2

从节点B到节点A的距离：3

new_cost = 2+3
= 5

[illegible]

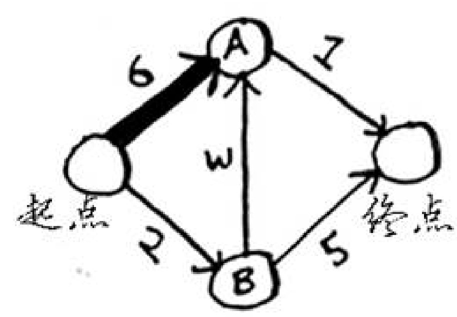
```
if costs[n] > new_cost:
```

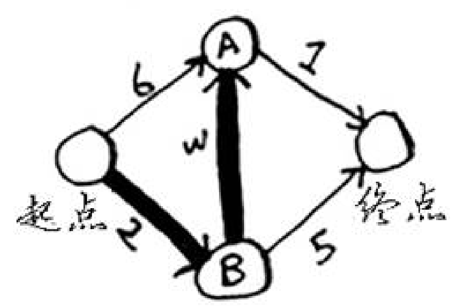
A	6
B	2

COSTS

原来前往节点A的开销为6

经节点B前往节点A的开销为5





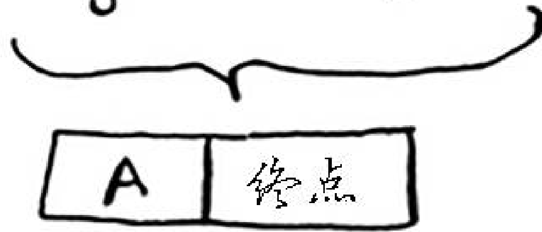
[illegible]A[illegible]A[illegible]

[illegible]B[illegible]A[illegible]B[illegible]

[illegible]`for`[illegible]

```
for n in neighbors.Keys():
```

n为“终点”



□□□B□□□□□□□□□□□□

```
new_cost = cost + neighbors[n]
```

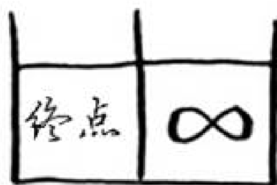
2

节点B到终点的距离：5

2+5
=7

□□7□□□□□□□□□□□□□□□□7□□□□

```
if costs[n] > new_cost:
```



在此之前，我们不知道前往终点的开销

7

□□□□□□□□□□□□□□

```
costs[n] = new_cost
```

"终点" → n, 7 → new_cost

A	5
B	2
终点	7

COSTS

```
parents[n] = node
```

"终点" → n, "B" → node

A	B
B	起点
终点	B

PARENTS

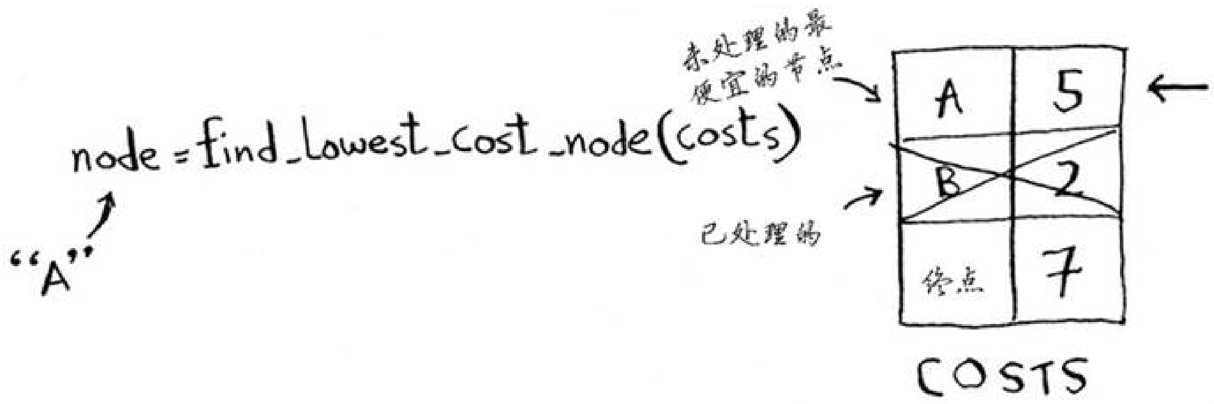
[illegible]B[illegible]B[illegible]

```
processed.append(node)
```

"B" → node

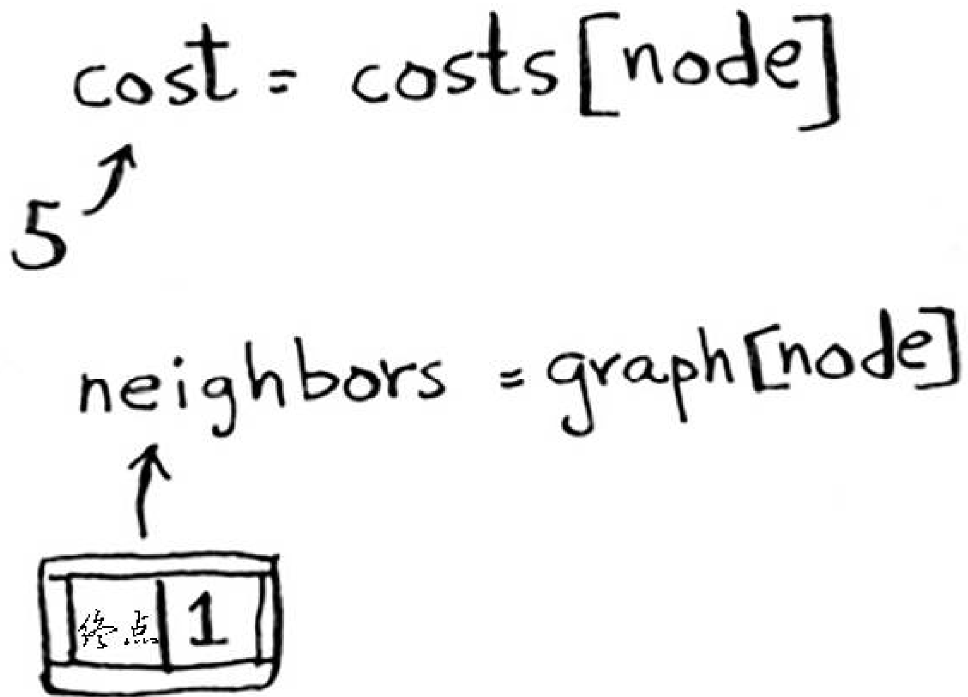
处理过的节点 B

[illegible]



未处理的最
便宜的节点
node = find_lowest_cost_node(costs)
"A"
已处理的
A
5
B
2
终点
7
COSTS

□□□□A□□□□□□□



cost = costs[node]
5
neighbors = graph[node]
终点
1

□□A□□□□□□□□□□□□

for n in neighbors.keys():

"终点" → n

neighbors.keys() → 终点

[illegible]7[illegible]A[illegible]

new_cost = cost + neighbors[n]

cost：从起点到节点A的开销：5

neighbors[n]：从节点A到终点的距离：1

5 + 1 = 6

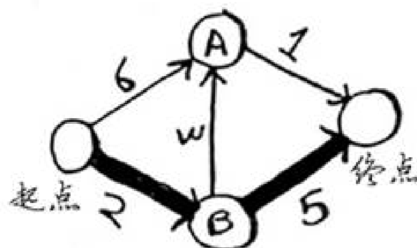
if costs[n] > new_cost:

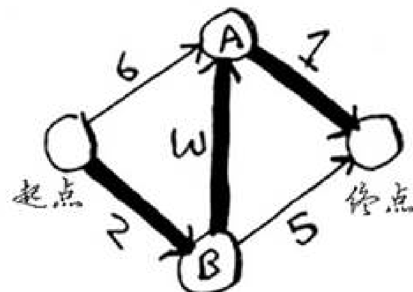
costs[n]：原来到终点的开销：7

new_cost：经节点A到终点的开销：6

D	2
终点	7

COSTS





[illegible]A[illegible]

[illegible] find_lowest_cost_node [illegible]

```
def find_lowest_cost_node(costs):
    lowest_cost = float("inf")
    lowest_cost_node = None
    for node in costs:  ←------[illegible]
        cost = costs[node]
        if cost < lowest_cost and node not in processed:  ←------[illegible]
            lowest_cost = cost  ←------[illegible]
            lowest_cost_node = node
    return lowest_cost_node
```

7.1







7.6

- [illegible]
- [illegible]
- [illegible]-[illegible]

[illegible] 8 [illegible] [illegible]

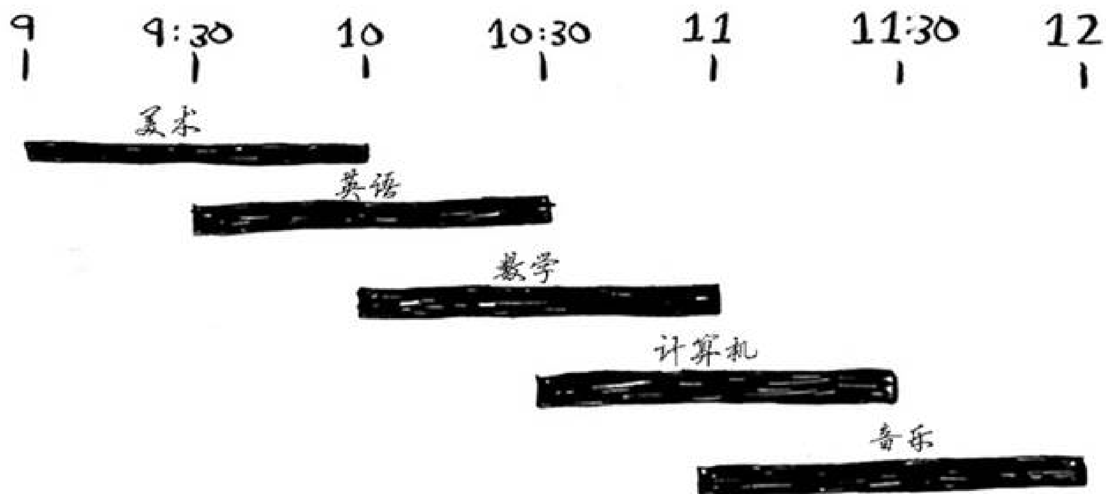
[illegible]

- [illegible]NP[illegible]
- [illegible]NP[illegible]
- [illegible]NP[illegible]
- [illegible]——[illegible]

8.1 [illegible]



课程	开始时间	结束时间
美术	9 AM	10 AM
英语	9:30AM	10:30AM
数学	10 AM	11 AM
计算机	10:30AM	11:30AM
音乐	11 AM	12 PM



(1)

(2) □□
□□

□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□10:00 a.m.□□□□□□□□□□□

美术	9AM	10AM	✓
英语	9:30AM	10:30AM	
数学	10AM	11AM	
计算机	10:30AM	11:30AM	
音乐	11AM	12PM	

□□□□□□□□10:00 a.m.□□□□□□□□□□□

美术	9AM	10AM	✓
英语	9:30AM	10:30AM	✗
数学	10AM	11AM	✓
计算机	10:30AM	11:30AM	
音乐	11AM	12PM	

美术	9AM	10AM	✓
英语	9:30AM	10:30AM	✗
数学	10AM	11AM	✓
计算机	10:30AM	11:30AM	✗
音乐	11AM	12PM	✓

[illegible]——[illegible]

[illegible]

8.2 [illegible]

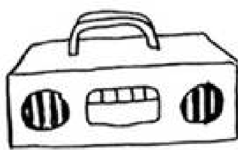
[illegible]35[illegible]1[illegible]≈0.45[illegible]

[illegible]

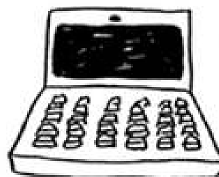


(1)

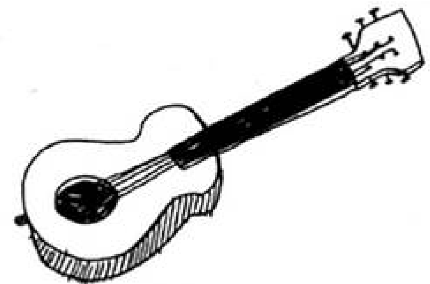
(2)



音响
3000美元
30磅

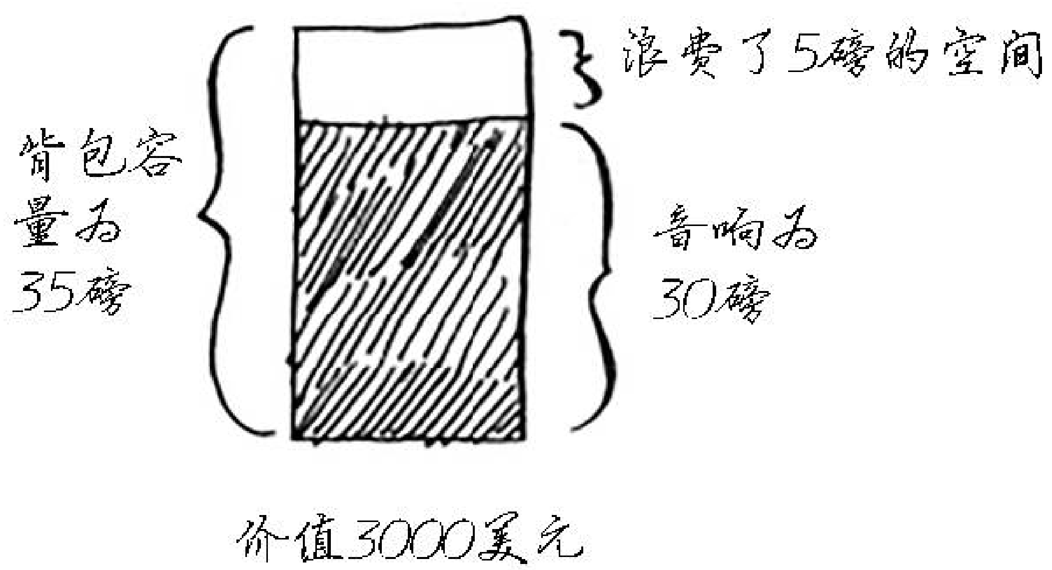


笔记本电脑
2000美元
20磅

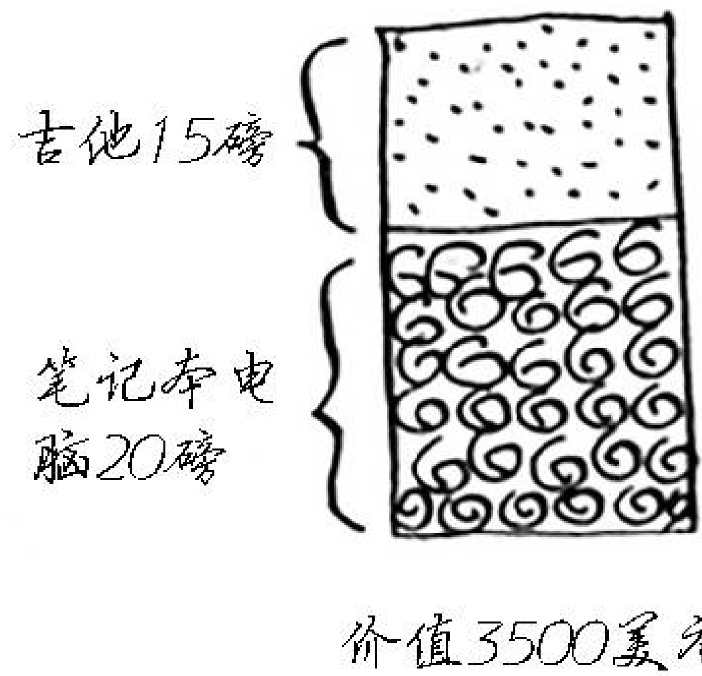


吉他
1500美元
15磅

35


浪费了5磅的空间
背包容量为35磅
音响为30磅
价值3000美元

□□□□□□3000□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□3500□□□


吉他15磅
笔记本电脑20磅
价值3500美元

□□□
□□□□□□□□□□□□□□□□□□□□□□□□

□□□
□□

[illegible]

8.1 [illegible]

8.2 [illegible]7[illegible]——[illegible]

[illegible]

8.3 [illegible]

[illegible]50[illegible]

广播台	覆盖的州
KONE	ID, NV, UT
KTWO	WA, ID, MT
KTHREE	OR, NV, CA
KFOUR	NV, UT
KFIVE	CA, AZ

……等等……

[illegible]50[illegible]

(1) [illegible]power set[illegible]2^n[illegible]



(2) [illegible]50[illegible]

[illegible]2^n[illegible]$O(2^n)$[illegible]5[illegible]10[illegible]10[illegible]

广播台数量	需要的时间
5	3.2 秒
10	102.4 秒
32	13.6 年
100	4×10^{21} 年

[illegible] [illegible]

[illegible]

[illegible]

(1) [illegible]

(2) [illegible]

[illegible] （approximation algorithm[illegible]

- [illegible]
- [illegible]

[illegible] $O(n^2)$[illegible] n [illegible]

[illegible]

1. [illegible]

[illegible]

[illegible]

```
states_needed = set(["mt", "wa", "or", "id", "nv", "ut",
"ca", "az"])  ←------[illegible]
```

[illegible]

```
>>> arr = [1, 2, 2, 3, 3, 3]
```

□□□□□□□□□□□□

```
>>> set(arr)
set([1, 2, 3])
```

□□□□□□□1□2□3□□□□□□□

[1,2,2,3,3,3] → 转换为集合 → (1,2,3)

集合

□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□

```
stations = {}
stations["kone"] = set(["id", "nv", "ut"])
stations["ktwo"] = set(["wa", "id", "mt"])
stations["kthree"] = set(["or", "nv", "ca"])
stations["kfour"] = set(["nv", "ut"])
stations["kfive"] = set(["ca", "az"])
```

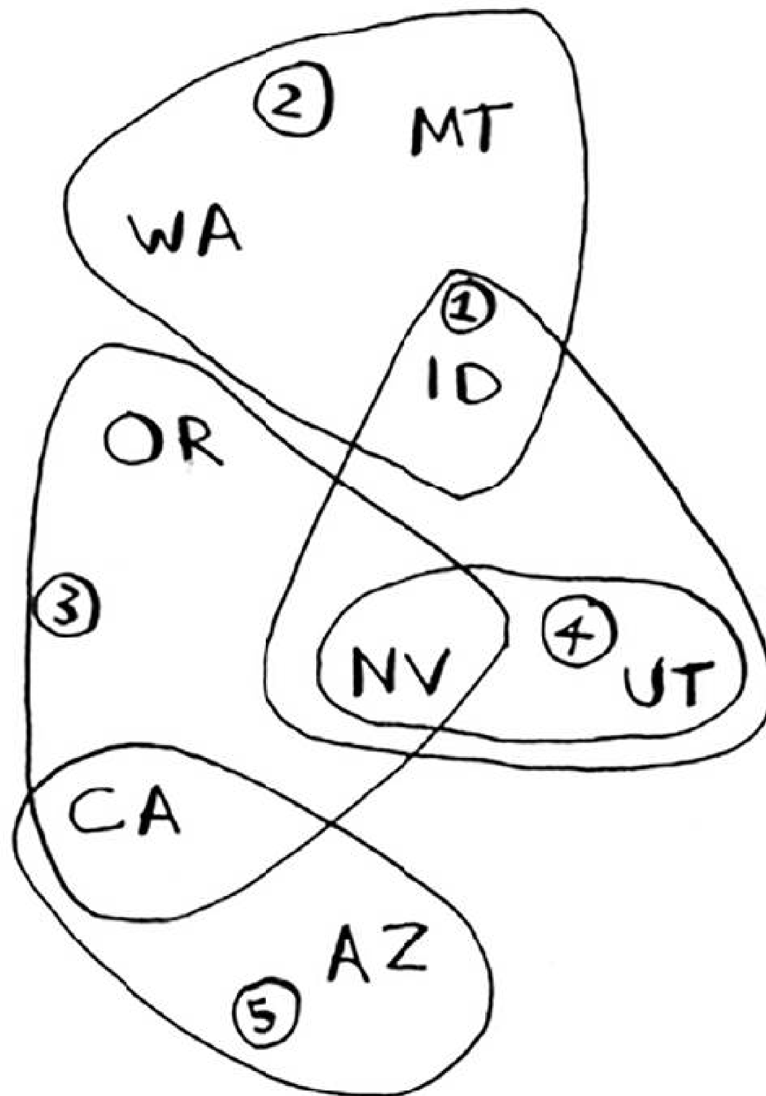
□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□kone□□□□□□□□□□□□□□□□
□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□

□□□□□□□□□□□□□□□□□□□□□□□□□

```
final_stations = set()
```

2. □□□□

□□



[illegible] best_station [illegible]

```
best_station = None
states_covered = set()
for station, states_for_station in stations.items():
```

states_covered [illegible] for [illegible] for [illegible]

```
covered = states_needed & states_for_station   ←-------[illegible]
if len(covered) > len(states_covered):
  best_station = station
  states_covered = covered
```

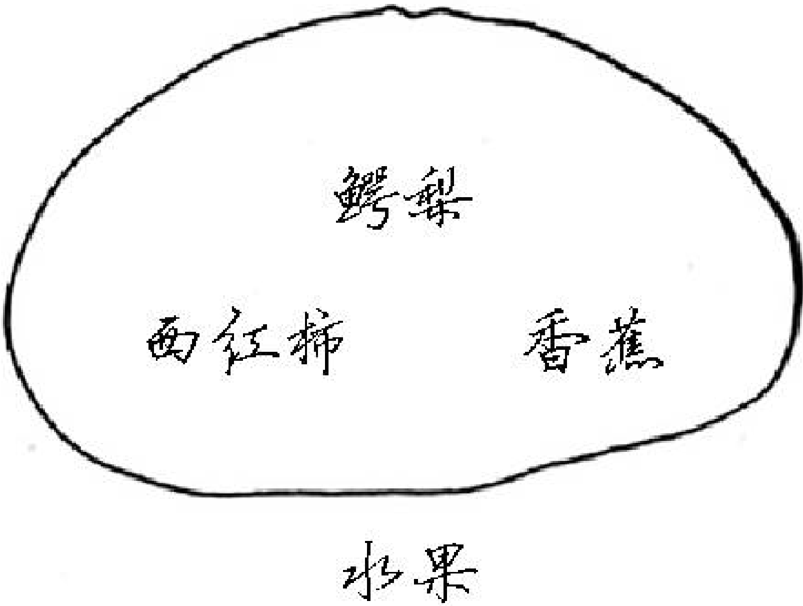
[illegible]

```
covered = states_needed & states_for_station
```

[illegible]

3. [illegible]

[illegible]



[illegible]

甜菜 胡萝卜

西红柿

蔬菜

[illegible]

[illegible]

属于水果或蔬菜的

鳄梨

甜菜 胡萝卜

西红柿 香蕉

并集

既是水果又是蔬菜的

鳄梨

香蕉

西红柿

甜菜

胡萝卜

交集



- [illegible]
- [illegible]
- [illegible]

[illegible]

```
>>> fruits = set(["avocado", "tomato", "banana"])
>>> vegetables = set(["beets", "carrots", "tomato"])
>>> fruits | vegetables  ←-------[illegible]
set(["avocado", "beets", "carrots", "tomato", "banana"])
>>> fruits & vegetables  ←-------[illegible]
set(["tomato"])
>>> fruits – vegetables  ←-------[illegible]
set(["avocado", "banana"])
>>> vegetables – fruits  ←-------[illegible]
```

[illegible]

- [illegible]
- [illegible]

4. [illegible]

[illegible]

[illegible]

```
covered = states_needed & states_for_station
```

covered [illegible]states_needed [illegible] states_for_station [illegible]best_station [illegible]

```
if len(covered) > len(states_covered):
  best_station = station
  states_covered = covered
```

□□□□□□□□□best_station □□□□□□□□□□□□□□for □□□□□□
best_station □□□□□□□□□□□□□

```
final_stations.add(best_station)
```

□□□□□states_needed □□□□□□□□□□□□□□□□□□□□□□□□□□

```
states_needed -= states_covered
```

□□□□□□□□□states_needed □□□□□□□□□□□□□□□

```
while states_needed:
  best_station = None
  states_covered = set()
  for station, states in stations.items():
    covered = states_needed & states
    if len(covered) > len(states_covered):
      best_station = station
      states_covered = covered


    states_needed -= states_covered
    final_stations.add(best_station)
```

□□□□□□□□final_stations □□□□□□□□□□□□□□□

```
>>> print final_stations
set(['ktwo', 'kthree', 'kone', 'kfive'])
```

[illegible]2[illegible]3[illegible]4[illegible]5[illegible]1[illegible]2[illegible]3[illegible]5[illegible]

	$O(n!)$	$O(n^2)$
广播台数量	精确算法	贪婪算法
5	3.2秒	2.5秒
10	102.4秒	10秒
32	13.6年	102.4秒
100	4×10^{21}年	16.67分钟

[illegible]

[illegible]

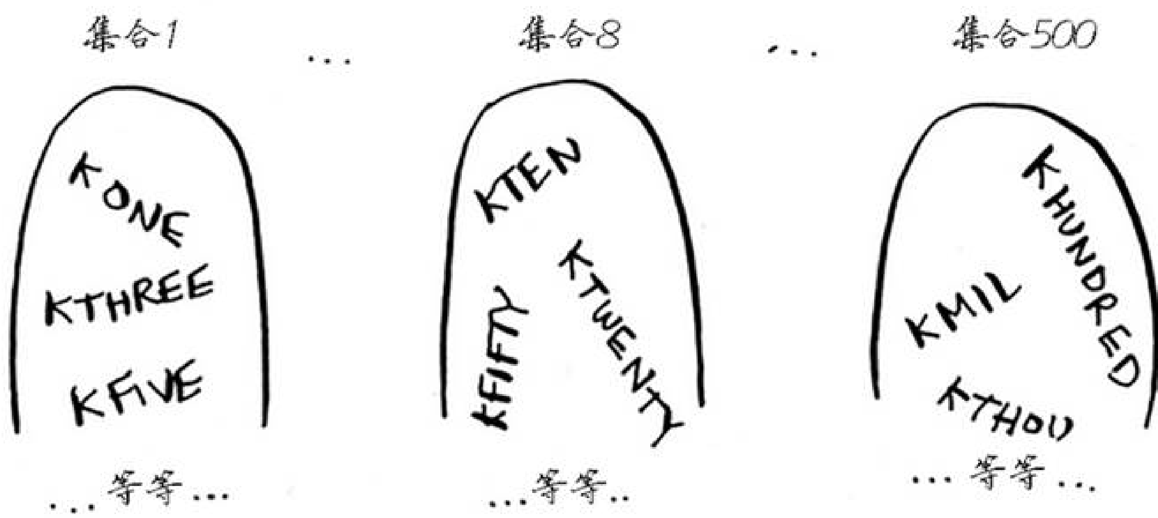
8.3 [illegible]

8.4 [illegible]

8.5 [illegible]

8.4 NP[illegible]

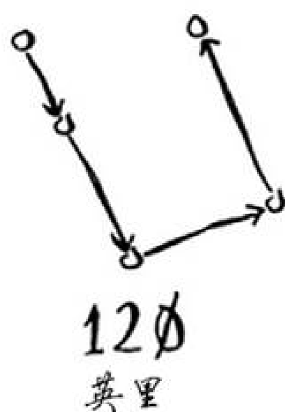
[illegible]

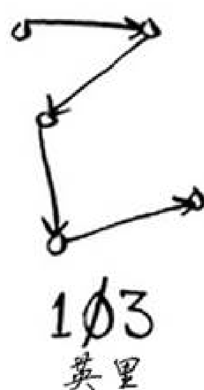

集合1
...
集合8
...
集合500
KONE
KTHREE
KFIVE
KTEN
KFIFTY
KTWENTY
KMIL
KHUNDRED
KTHOU
...等等...
...等等..
...等等...

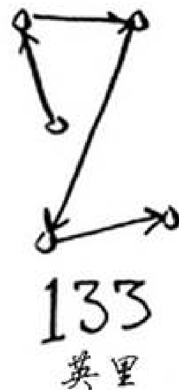
[illegible]1[illegible]5[illegible]


马林
伯克利
旧金山
弗里蒙特
帕洛
阿尔托

[illegible]5[illegible]



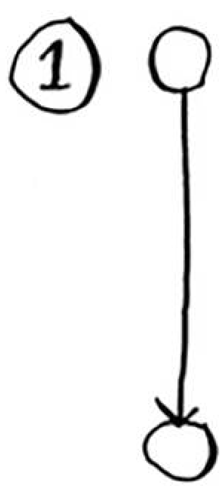




[illegible]5[illegible]

8.4.1 [illegible]

[illegible]





[illegible]

[illegible]

[illegible]4[illegible]

[illegible]50[illegible]

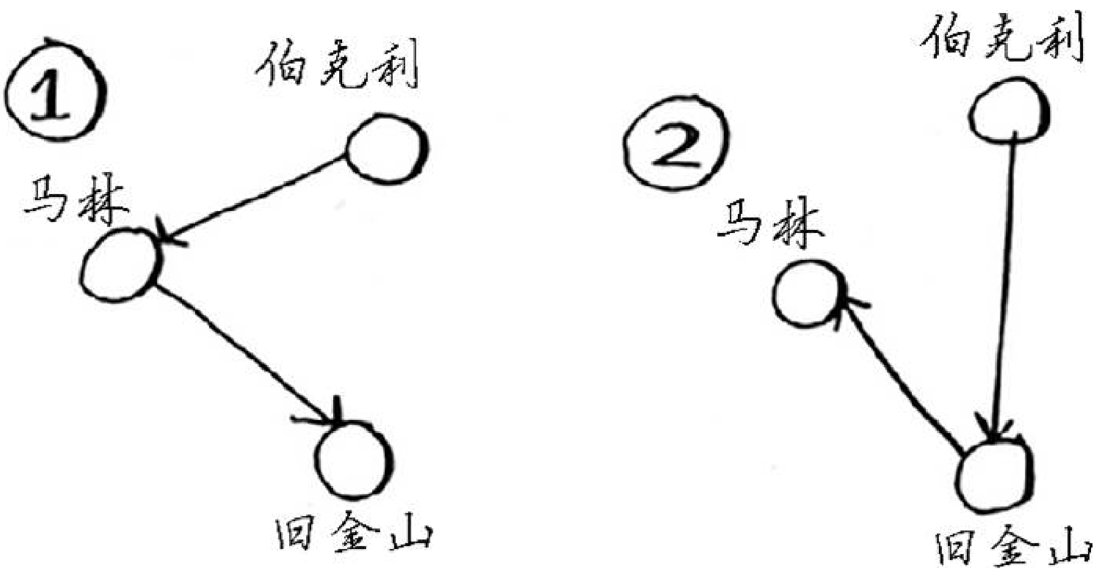
[illegible]

[illegible]

1. 3[illegible]

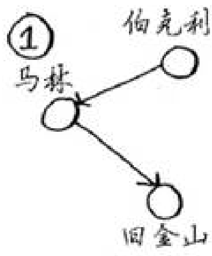
[illegible]

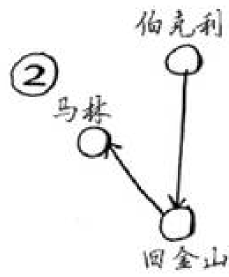
[illegible]



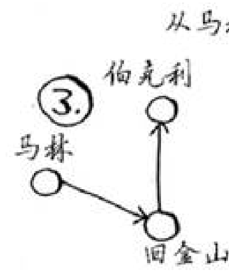
[illegible]6[illegible]

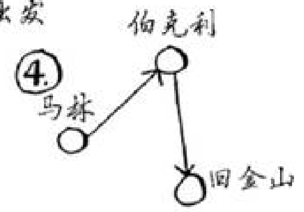
从伯克利出发



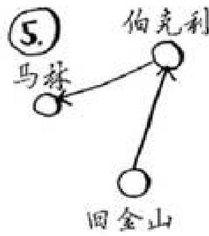


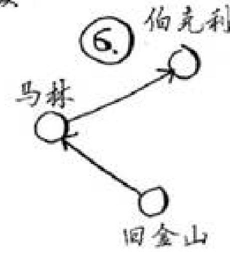
从马林出发





从旧金山出发

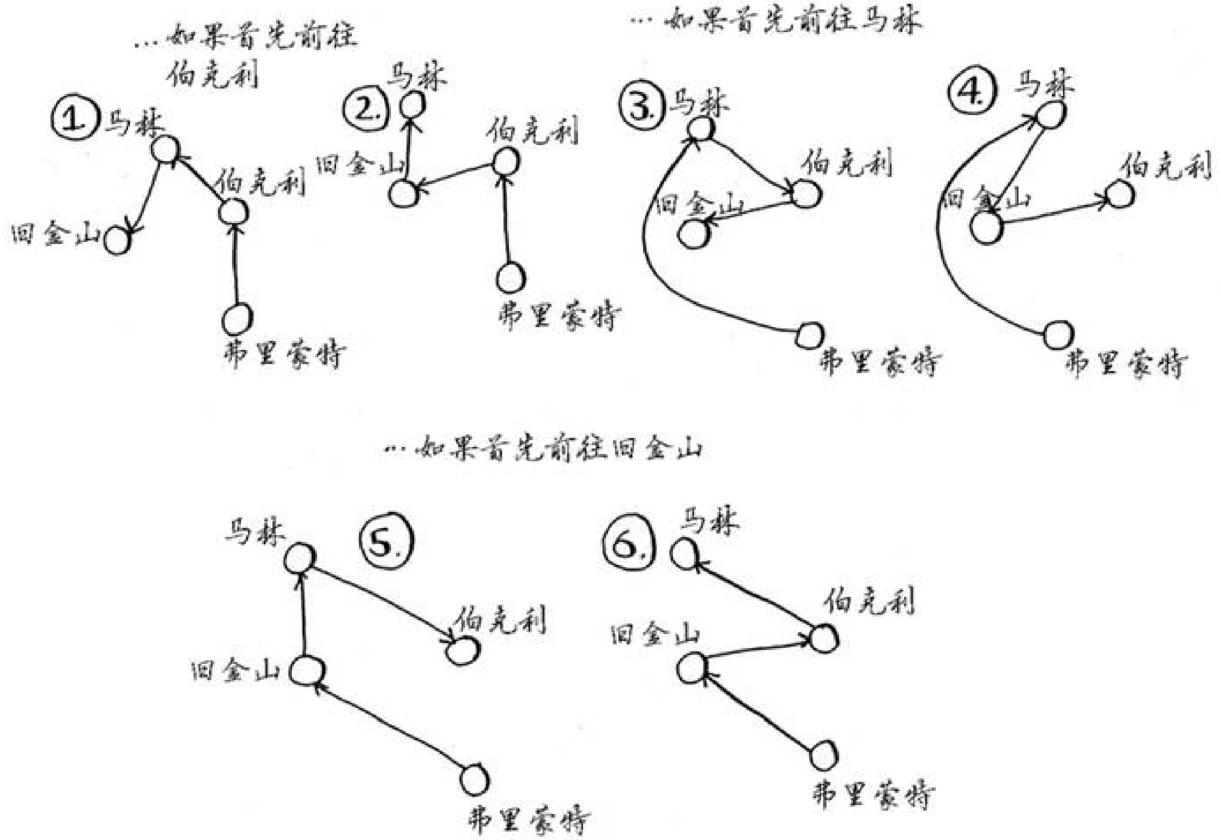




[illegible]3[illegible]6[illegible]

2. **4**[illegible]

[illegible]——[illegible]



从弗里蒙特出发
…如果首先前往伯克利
1. 马林
伯克利
旧金山
弗里蒙特
2. 马林
旧金山
伯克利
弗里蒙特
…如果首先前往马林
3. 马林
伯克利
旧金山
弗里蒙特
4. 马林
伯克利
旧金山
弗里蒙特
…如果首先前往旧金山
马林
5.
伯克利
旧金山
弗里蒙特
6. 马林
伯克利
旧金山
弗里蒙特

从马林出发

= 有6条可能的路线 =

从旧金山出发

= 有6条可能的路线 =

从伯克利出发

= 有6条可能的路线 =

[illegible]4[illegible]6[illegible]4 × 6 = 24[illegible]

[illegible]

城市数

1 → 1条路线

2 → 2个可能的出发城市 × 每个出发城市1条可能的路线 = 2条可能的路线

3 → 3个可能的出发城市 × 每个出发城市2条可能的路线 = 6条可能的路线

4 → 4个可能的出发城市 × 每个出发城市6条可能的路线 = 24条可能的路线

5 → 5个可能的出发城市 × 每个出发城市24条可能的路线 = 120条可能的路线

[illegible]6[illegible]720[illegible]7[illegible]5040[illegible]8[illegible]40 320[illegible]

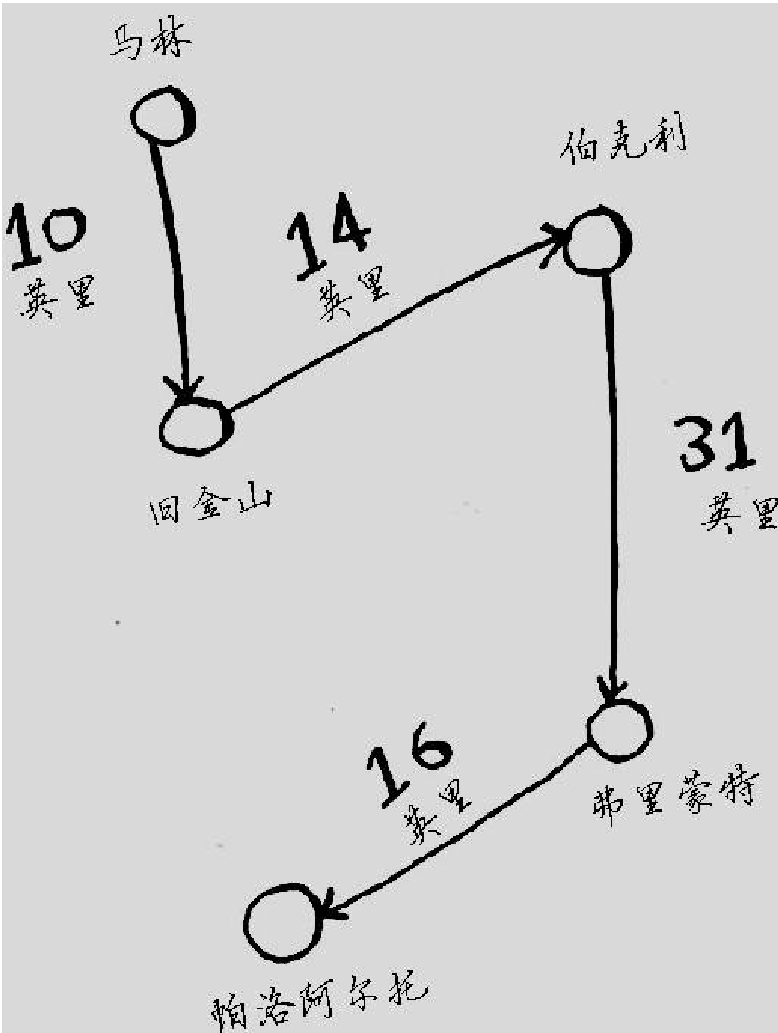
[illegible]（factorial function[illegible]3[illegible]5! = 120[illegible]10[illegible]10! = 3 628 800[illegible]10[illegible]300[illegible]

[illegible]/[illegible]NP[illegible]

[illegible]

[illegible]

[illegible]



[illegible]71[illegible]

NP[illegible]

8.4.2 [illegible]NP[illegible]



Jonah[illegible]

球员	能力
MATT FORTE	跑卫
BRENDAN MARSHALL	外接手 / 能够承受压力
AARON RODGERS	四分卫 / 能够承受压力
...	...

Jonah[illegible]Jonah[illegible]

Jonah[illegible]

(1) [illegible]

(2) [illegible]

NP[illegible]NP[illegible]NP[illegible]NP[illegible]A[illegible]B[illegible]

[illegible]——NP[illegible]
[illegible]NP[illegible]

- [illegible]
- [illegible]“[illegible]”[illegible]NP[illegible]
- [illegible]NP[illegible]
- [illegible]NP[illegible]
- [illegible]NP[illegible]
- [illegible]NP[illegible]

[illegible]

8.6 [illegible]20[illegible]20[illegible]NP[illegible]

8.7 [illegible]NP[illegible]

8.8 [illegible]NP[illegible]

8.5 [illegible]

- [illegible]
- [illegible]NP[illegible]
- [illegible]NP[illegible]
- [illegible]

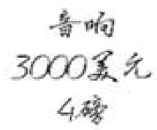
9

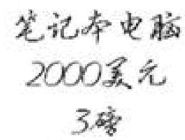
9.1

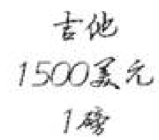
8 4



3


音响
3000美元
4磅

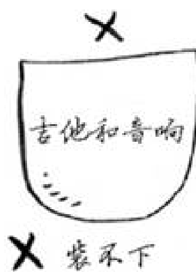

笔记本电脑
2000美元
3磅

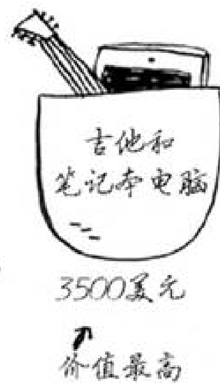

吉他
1500美元
1磅

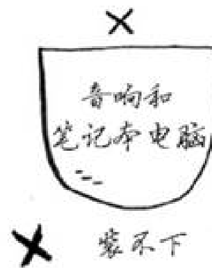
[illegible]

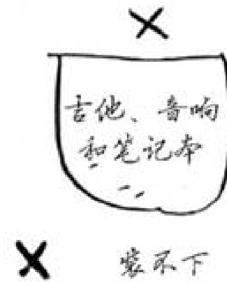
9.1.1 [illegible]

[illegible]

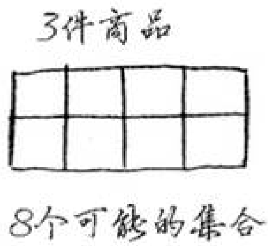

无
0
吉他
1500美元
音响
3000美元
笔记本电脑
2000美元
吉他和音响
装不下

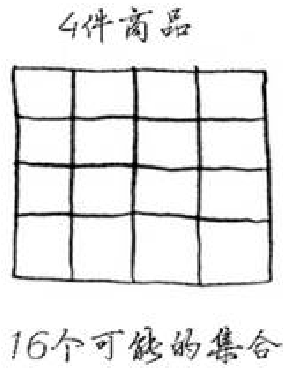

吉他和
笔记本电脑
3500美元
价值最高

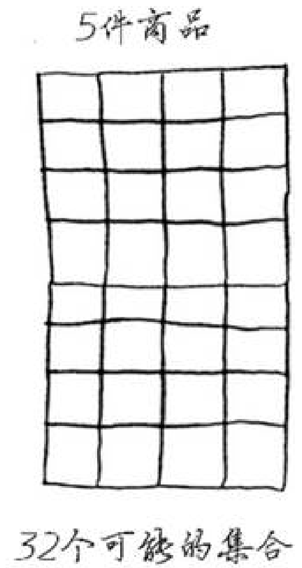

音响和
笔记本电脑
装不下


吉他、音响
和笔记本
装不下

□□□□□□□□□□□□□□3□□□□□□□□□□□□□8□□□□□□□□4□□□□□□□□□□16□
□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□$O(2^n)$□□□□□□□□□







如果有32件商品
将大约有40亿个
可能的集合!

□□□□□□□□□□□□□□□□□□□□□□□□□8□□□□□□□□□□□□□□□□□□□□□□□□□□
□□□□

□□□□□□□□□□□

9.1.2 □□□□

□□

□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□

1磅

\+

3磅

=

1磅

3磅

4磅

[illegible]

[illegible]

[illegible]

各列为不同容量（1~4磅）的背包

各行为可选择的商品

	1	2	3	4
吉他				
音响				
笔记本电脑				

[illegible]1[illegible]4[illegible]

[illegible]

1. [illegible]

[illegible]

	1	2	3	4
吉他				
音响				
笔记本电脑				

[illegible]

[illegible]1[illegible]1[illegible]1500[illegible]

[illegible]

	1	2	3	4
吉他（G）	$1500 G			
音响				
笔记本电脑				

[illegible]

[illegible]2[illegible]

	1	2	3	4
吉他（G）	$1500 G	$1500 G		
音响				
笔记本电脑				

[illegible]

	1	2	3	4
吉他（G）	$1500 G	$1500 G	$1500 G	$1500 G
音响				
笔记本电脑				

[illegible]4[illegible]1[illegible]2[illegible]

[illegible]

	1	2	3	4
吉他（G）	$1500 G	$1500 G	$1500 G	$1500 G
音响				
笔记本电脑				

[illegible]4[illegible]1500[illegible]

	1	2	3	4
吉他（G）	$1500 G	$1500 G	$1500 G	$1500 G
音响				
笔记本电脑				

当前，为了让背包中商品的价值最高，小偷应盗窃价值1500美元的吉他

[illegible]

2. [illegible]

[illegible]——[illegible]1[illegible]1[illegible]1500[illegible]

背包容量为1磅时，之前可装入的商品的最大价值

	1	2	3	4
吉他（G）	$1500 G	$1500 G	$1500 G	$1500 G
音响				
笔记本电脑				

背包容量为1磅时，现在可装入的商品的最大价值

[illegible]

[illegible]1[illegible]1[illegible]
1500[illegible]

	1	2	3	4
吉他（G）	$1500 G	$1500 G	$1500 G	$1500 G
音响	$1500 G			
笔记本电脑				

[illegible]2[illegible]3[illegible]
1500[illegible]

	1	2	3	4
吉他（G）	$1500 G	$1500 G	$1500 G	$1500 G
音响	$1500 G	$1500 G	$1500 G	
笔记本电脑				

[illegible]

[illegible]4[illegible]1500[illegible]
[illegible]3000[illegible]

	1	2	3	4
吉他（G）	$1500 G	$1500 G	$1500 G	$1500 G
音响（S）	$1500 G	$1500 G	$1500 G	$3000 S
笔记本电脑				

[illegible]4[illegible]3000[illegible]

	1	2	3	4	
吉他（G）	$1500 G	$1500 G	$1500 G	$1500 G	← 以前的最大价值
音响（S）	$1500 G	$1500 G	$1500 G	$3000 S	← 最新的最大价值
笔记本电脑					← 最终解

3. [illegible]

[illegible]3[illegible]1[illegible]2[illegible]1500[illegible]

	1	2	3	4
吉他（G）	$1500 G	$1500 G	$1500 G	$1500 G
音响（S）	$1500 G	$1500 G	$1500 G	$3000 S
笔记本电脑	$1500 G	$1500 G		

[illegible]3[illegible]1500[illegible]2000[illegible]
[illegible]2000[illegible]

	1	2	3	4
吉他（G）	$1500 G	$1500 G	$1500 G	$1500 G
音响（S）	$1500 G	$1500 G	$1500 G	$3000 S
笔记本电脑（L）	$1500 G	$1500 G	$2000 L	

[illegible]4[illegible]3000[illegible]
[illegible]2000[illegible]

$3000 vs $2000

音响　　笔记本电脑

[illegible]3[illegible]1[illegible]

$3000 vs ($2000 + ???)

音响　　笔记本电脑　　余下的1磅容量

[illegible]1[illegible]

1磅容量可装入商品的最大价值

1	2	3	4
$1500 G	$1500 G	$1500 G	$1500 G
$1500 G	$1500 G	$1500 G	$3000 S
$1500 G	$1500 G	$2000 L	

[illegible]1[illegible]1500[illegible]

$3000 vs ($2000 + $1500)

音响 笔记本电脑 吉他

[illegible]3500[illegible]

[illegible]

	1	2	3	4
吉他（G）	$1500 G	$1500 G	$1500 G	$1500 G
音响（S）	$1500 G	$1500 G	$1500 G	$3000 S
笔记本电脑（L）	$1500 G	$1500 G	$2000 L	$3500 L G

最终答案

[illegible]3500[illegible]

[illegible]

CELL[i][j]（行 列）= 两者中较大的那个：

1. 上一个单元格的值（即 CELL[i-1][j] 的值）

VS

2. 当前商品的价值 + 剩余空间的价值（CELL[i-1][j - 当前商品的重量]）

[illegible]

1磅 + 3磅 = 4磅

9.2 [illegible]FAQ

[illegible]

9.2.1 [illegible]

[illegible]——[illegible]iPhone[illegible]

IPHONE

2000美元

1磅

[illegible]

	1	2	3	4
吉他（G）	\$1500 G	\$1500 G	\$1500 G	\$1500 G
音响（S）	\$1500 G	\$1500 G	\$1500 G	\$3000 S
笔记本电脑（L）	\$1500 G	\$1500 G	\$2000 L	\$3500 LG

□□□□□□□□□4□□□□□□□□□□3500□□□□□□□□□□□□□□□□□□□□□
iPhone□□□

	1	2	3	4
吉他（G）	$1500 G	$1500 G	$1500 G	$1500 G
音响（S）	$1500 G	$1500 G	$1500 G	$3000 S
笔记本电脑（L）	$1500 G	$1500 G	$2000 L	$3500 LG
iPhone				

新的答案

□□□□□□□□□□□□□□□□□□□□□□□□□□□□

□□□□□□□□□□□iPhone□□□□□1□□□□□□□□□□□□1500□□□□
iPhone□□2000□□□□□□iPhone□□□□□

	1	2	3	4
吉他（G）	$1500 G	$1500 G	$1500 G	$1500 G
音响（S）	$1500 G	$1500 G	$1500 G	$3000 S
笔记本电脑（L）	$1500 G	$1500 G	$2000 L	$3500 LG
iPhone（I）	$2000 I			

[illegible]iPhone[illegible]

$1500 G	$1500 G	$1500 G	$1500 G
$1500 G	$1500 G	$1500 G	$3000 S
$1500 G	$1500 G	$2000 L	$3500 LG
$2000 I	$3500 IG		

□□□□□□□□□□□□□□□□iPhone□□□□□□□□□□

□□□□□□□□□□□□□□□□□□□□□□□□□□3500□□□□□□□iPhone□□□□□3□□□□□

3□□□□□□□□□2000□□□□□□iPhone□□2000□□□□□□□4000□□□□□□□□□□□□□

□□□□□□□□

$1500 G	$1500 G	$1500 G	$1500 G
$1500 G	$1500 G	$1500 G	$3000 S
$1500 G	$1500 G	$2000 L	$3500 LG
$3500 I	$3500 IG	$3500 IG	$4000 IL

新答案

往下走时最大价值有可能下降吗?

	1	2	3	4
	$1500	$1500	$1500	$1500
	Ø	Ø	Ø	$3000

[illegible]

[illegible]

[illegible]

9.1 [illegible]——MP3[illegible]1[illegible]1000[illegible]

9.2.2 [illegible]

[illegible]

[illegible]

	1	2	3	4
音响（S）	Ø	Ø	Ø	$3000 S
笔记本电脑（L）	Ø	Ø	$2000 L	$3000 S
吉他（G）	$1500 G	$1500 G	$2000 L	$3500 LG

[illegible]

9.2.3 [illegible]

[illegible]

9.2.4 [illegible]

[illegible] 0.5 [illegible] 1000 [illegible] 3.5 [illegible] 3.5 [illegible] 1 [illegible] 2 [illegible] 3 [illegible] 4 [illegible] 3.5 [illegible]

[illegible]

	0.5	1	1.5	2	2.5	3	3.5	4
吉他								
音响								
笔记本电脑								
项链								

9.2.5

藜麦
每磅6美元

木豆
每磅3美元

大米
每磅2美元

9.2.6

名胜	时间	评分
威斯敏斯特教堂	0.5天	7
环球剧场	0.5天	6
英国国家美术馆	1天	9
大英博物馆	2天	9
圣保罗大教堂	0.5天	8

	1/2	1	1 1/2	2
威斯敏斯特教堂				
环球剧场				
英国国家美术馆				
大英博物馆				
圣保罗大教堂				

	1/2	1	1½	2
威斯敏斯特教堂（W）	7 W	7 W	7 W	7 W
环球剧场（G）	7 W	13 WG	13 WG	13 WG
英国国家美术馆（N）	7 W	13 WG	16 WN	22 WGN
大英博物馆（B）	7 W	13 WG	16 WN	22 WGN
圣保罗大教堂（S）	8 S	15 WS	21 WGS	24 WNS

最终的答案：去游览威斯敏斯特教堂、英国国家美术馆和圣保罗大教堂

9.2.7 [illegible]

[illegible]

埃菲尔铁塔	1.5天	8
卢浮宫	1.5天	9
巴黎圣母院	1.5天	7

[illegible]3[illegible]4.5[illegible]

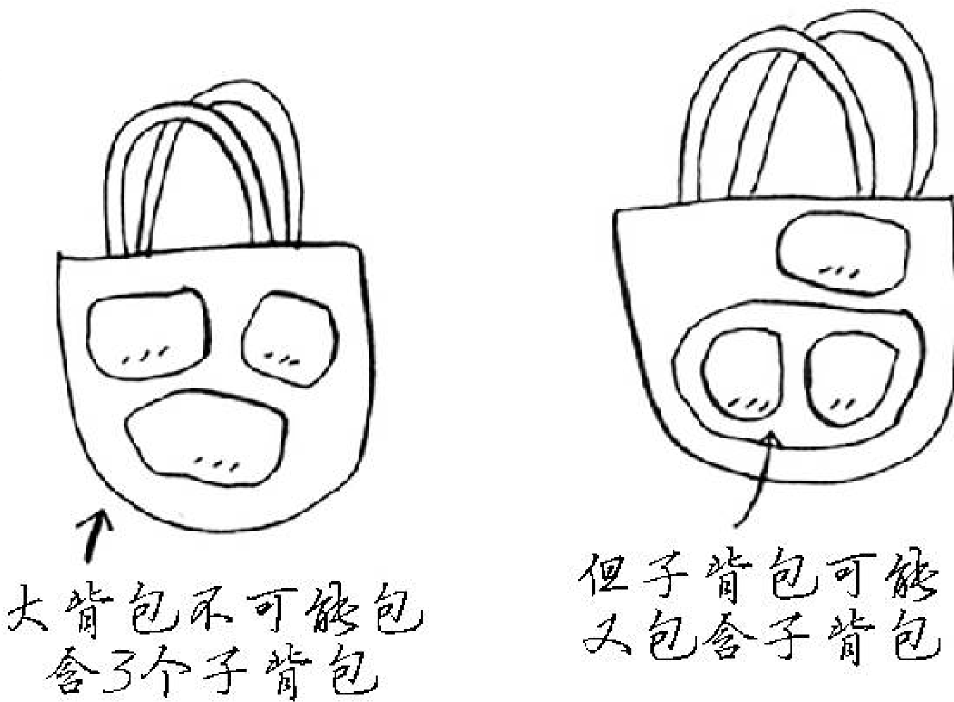
[illegible]1[illegible]3[illegible]3.5[illegible]1[illegible]4.5[illegible]

□□□□□□□□“□□”□□□□□□□“□□”□□□1□□□□□□□1.5□□□□□□□□□□□□□□□
□□□□

□□ □□□□□□□□□□□□□□□□□□□□□□□□□

9.2.8 □□□□□□□□□□□□□□□□□□□□□

□□



9.2.9 □□□□□□□□□□□□□□□

□□□□□□□□□□□□□□□□□□□□



□□□□□□□□□□□3.5□□□□100□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□
□□□□□□□□□□□0.5□□□□□□□□□□□□

□□

9.2 □□□□□□□□□□□□□□□□6□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□
□□□□□□□□□□□□□□□□

- □□□3□□□□10□□
- □□□1□□□□3□
- □□□□2□□□□9□□
- □□□□2□□□□5□□
- □□□□1□□□□6□□

□□□□□□□□□□□□□□□

9.3 [illegible]

[illegible]



- [illegible]

- [illegible]

[illegible]

- [illegible]

- [illegible]

- [illegible]

[illegible]dictionary.com[illegible]

[illegible]Alex[illegible]fish[illegible]hish[illegible]



Alex[illegible]hish[illegible]fish[illegible]vista[illegible]

与FISH类似的单词:

- FISH
- VISTA

9.3.1 [illegible]

[illegible]

- [illegible]
- [illegible]

- [illegible]

[illegible]hish[illegible]fish[illegible]hish[illegible]vista[illegible]

[illegible]

[illegible]hish[illegible]fish[illegible]his[illegible]fis[illegible]

[illegible]

9.3.2 [illegible]

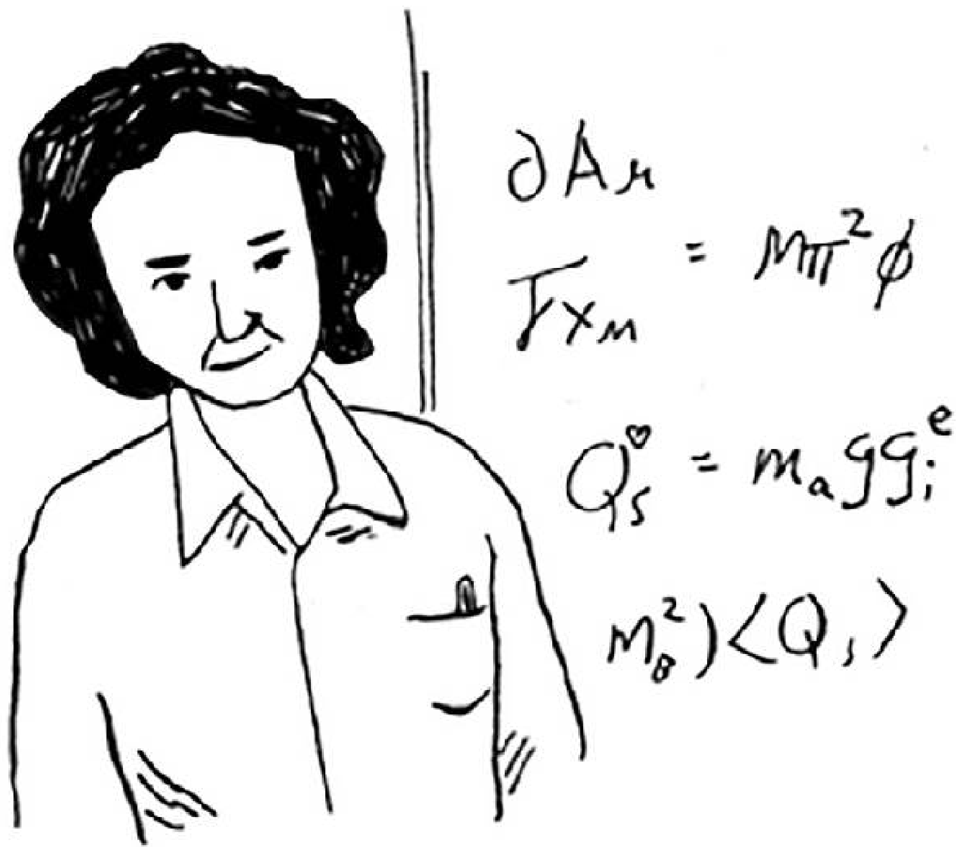
[illegible]——hish[illegible]fish[illegible]ish[illegible]

[illegible]（Feynman algorithm）[illegible]·[illegible]

(1) [illegible]

(2) [illegible]

(3) [illegible]



[illegible]

[illegible]

[illegible]

	H	I	S	H
F	0	0		
I				
S			2	0
H				3

□□□□□□□□□□□□□□□□□□□□□□□□□□ □□□□□□□□(3, 3)□□□2□□□□□□□□(3, 4)□□□0□□

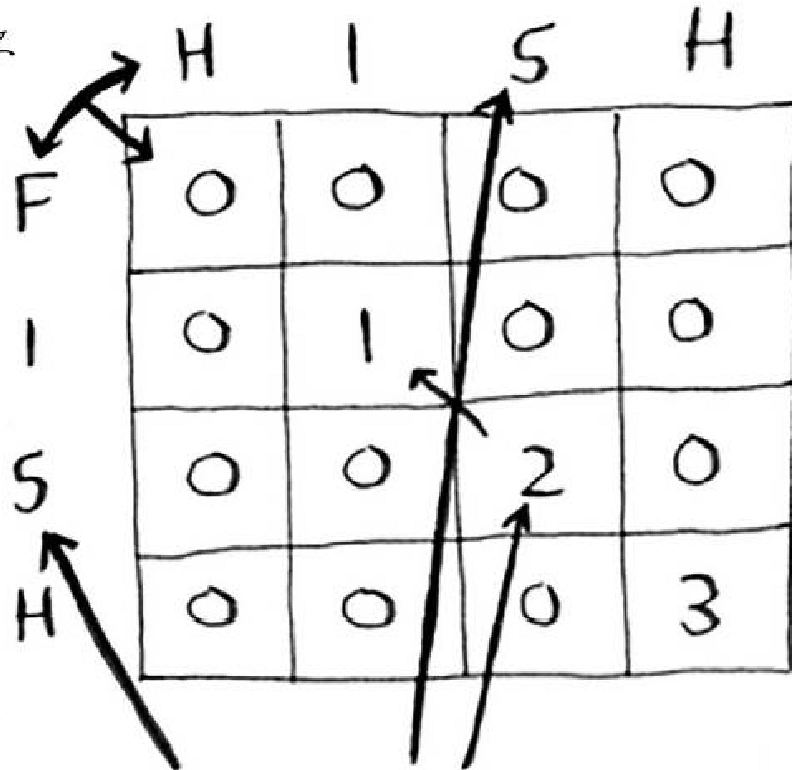
□□

9.3.3 □□□□

□□□□□□□□

	H	I	S	H
F	0	0	0	0
I	0	1	0	0
S	0	0	2	0
H	0	0	0	3

[illegible]



□□□□□□□□□□□□□□□□□□□□□□□

```
if word_a[i] == word_b[j]:  ←---------□□□□□□
  cell[i][j] = cell[i-1][j-1] + 1
else:  ←----------------------------□□□□□□
  cell[i][j] = 0
```

□□□□□hish□vista□□□□□□□□□□□□□□□□□□

[illegible]——[illegible]

[illegible]hish[illegible]hish[illegible]fish[illegible]hish[illegible]vista[illegible]

[illegible]Alex[illegible]fish[illegible]

9.3.4 [illegible]

[illegible]Alex[illegible]fosh[illegible]fish[illegible]fort[illegible]

[illegible]

	F	O	S	H
F	1	0	0	0
O	0	2	0	0
R	0	0	0	0
T	0	0	0	0

VS

	F	O	S	H
F	1	0	0	0
I	0	0	0	0
S	0	0	1	0
H	0	0	0	2

□□□□□□□□□□□□□□□□□□□□□fosh□fish□□□

F O S H
↓ ↓ ↓
F I S H = 3

F O S H
↓ ↓
F O R T = 2

□□□□□□□□□□□□ □□□□□□□□□□□□□□ □□□□□□□□□□□□□□□□□□□□□□□□
□□□□□□□

□□□□□□□fish□fosh□□□□□□□□□□□□□□□□

	F	O	S	H
F	1	1		
I	1			
S		1	2	2
H				

[illegible]——[illegible]

9.3.5 [illegible]

[illegible]

	F	O	S	H
F	1	1	1	1
O	1	2	2	2
R	1	2	2	2
T	1	2	2	2

最长公共子序列 =2

vs

	F	O	S	H
F	1	1	1	1
I	1	1	1	1
S	1	1	2	2
H	1	1	2	3

最长公共子序列 =3

[illegible]

1. 如果两个字母不同，就选择上方和左方邻居中较大的那个

（与计算最长公共子串时不同）

	F	O	S	H
F	1	1		
I	1	1		
S			2	
H				

2. 如果两个字母相同，就将当前单元格的值设置为左上方单元格的值加1（与计算最长公共子串时类似）

[illegible]

```
if word_a[i] == word_b[j]:      ←--------[illegible]
  cell[i][j] = cell[i-1][j-1] + 1
else:     ←------------------------------[illegible]
  cell[i][j] = max(cell[i-1][j], cell[i][j-1])
```

[illegible]

- [illegible] DNA [illegible]
- [illegible] git diff [illegible]
- [illegible] levenshtein distance [illegible]

- □□□□□□Microsoft Word□□□

□□

9.3 □□□□□□□□□□blue□clues□□□□□□□□□□

9.4 □□

- □□□□□□□□□□□□□□□□□□□□□□□□□□
- □□□□□□□□□□□□□□□□□□□□□□□□
- □□□□□□□□□□□□□□□□
- □□□□□□□□□□□□□□□□□
- □□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□
- □□□□□□□□□□□□□□□□□□□□□□□

□ 10 □ K□□□□□

□□□□

- □□□□K□□□□□□□□□□□□
- □□□□□□□
- □□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□
- □□K□□□□□□□□□□□□□□□

10.1 □□□□□



红色

颜色

橙色

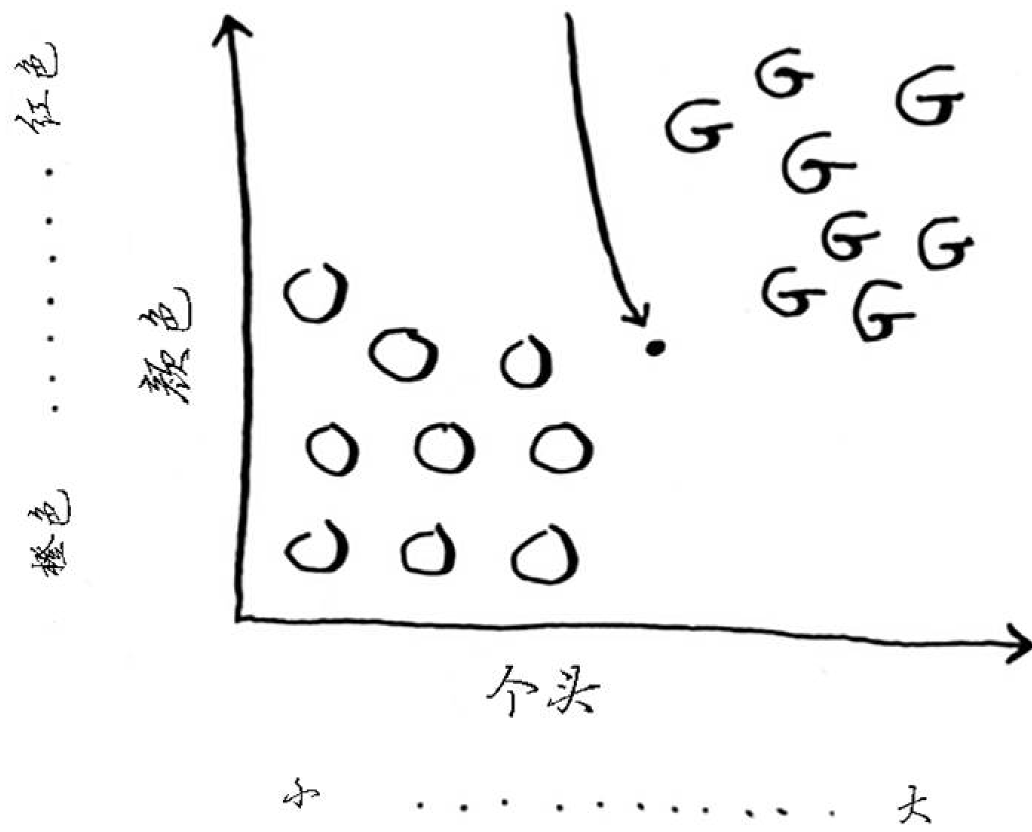
个头

小 大

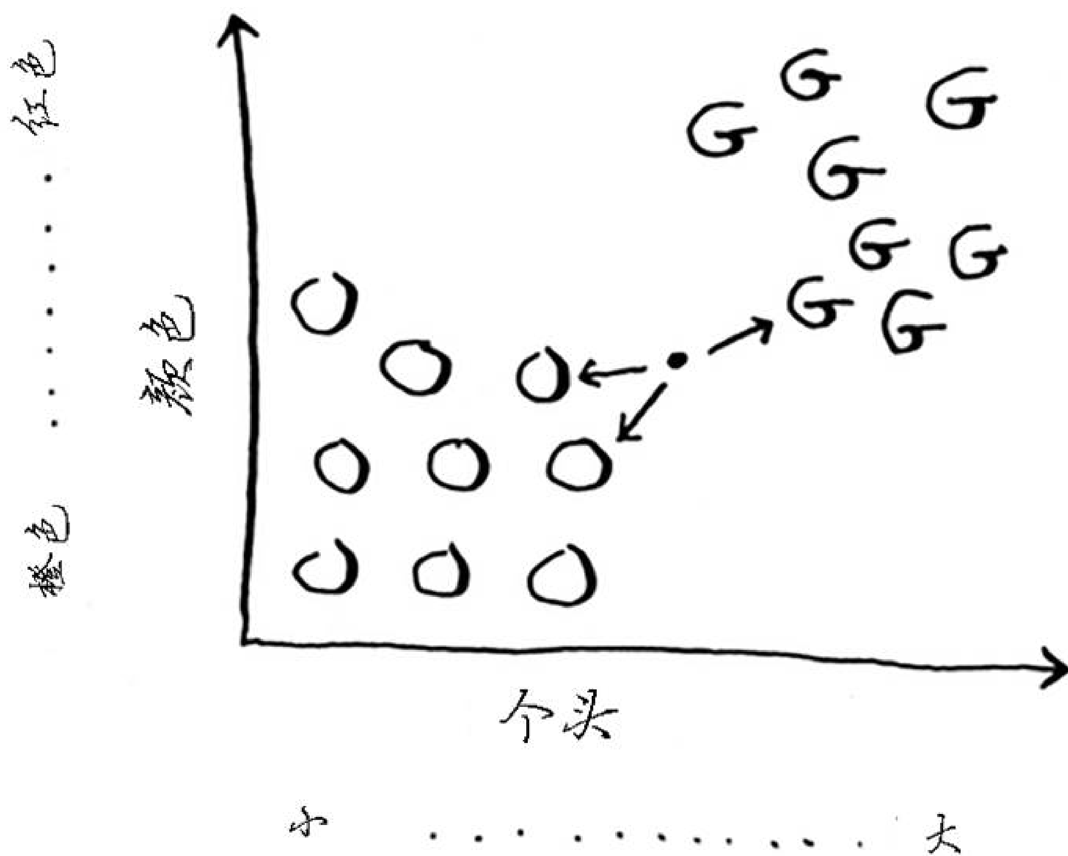
O表示橙子

G表示柚子

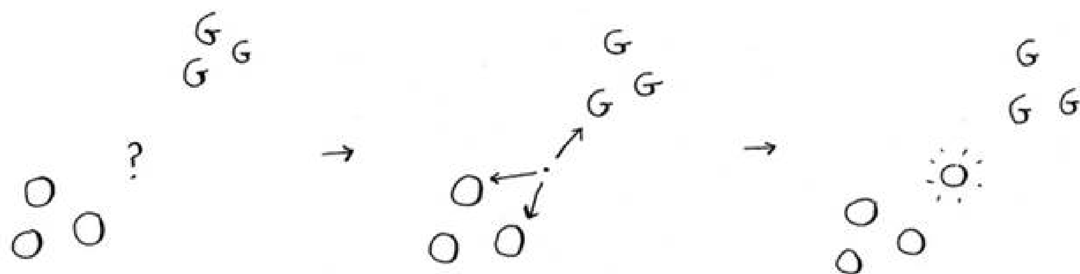
[illegible]



神秘的水果
红色
颜色
橙色
个头
小
大



[illegible]**K**[illegible]k-nearest neighbours[illegible]KNN[illegible]

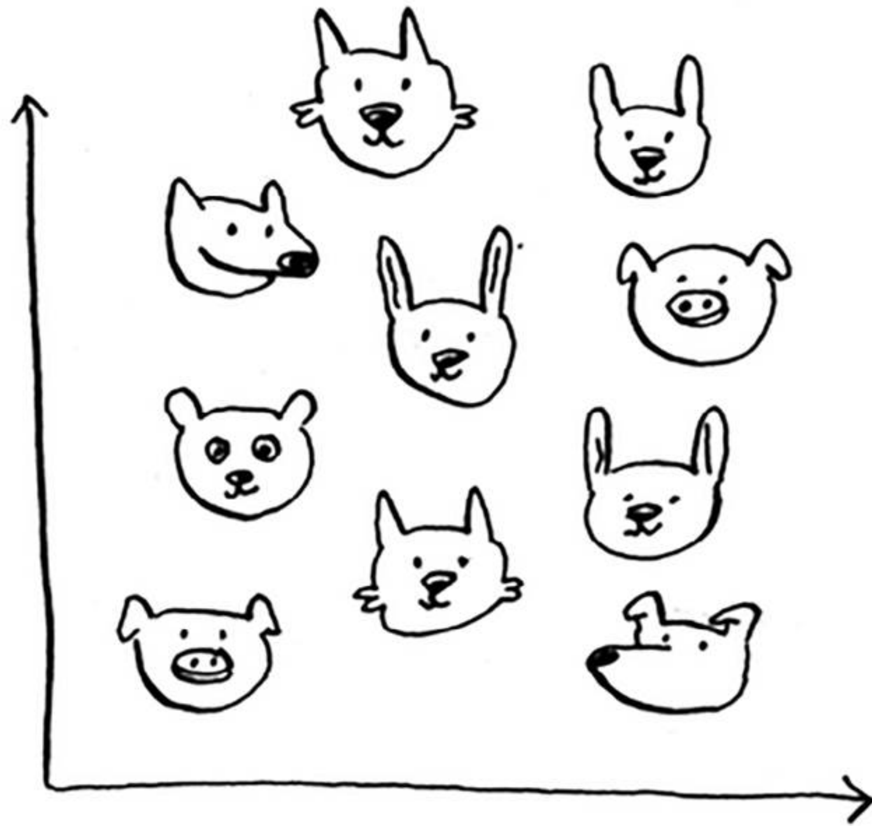


KNN[illegible]

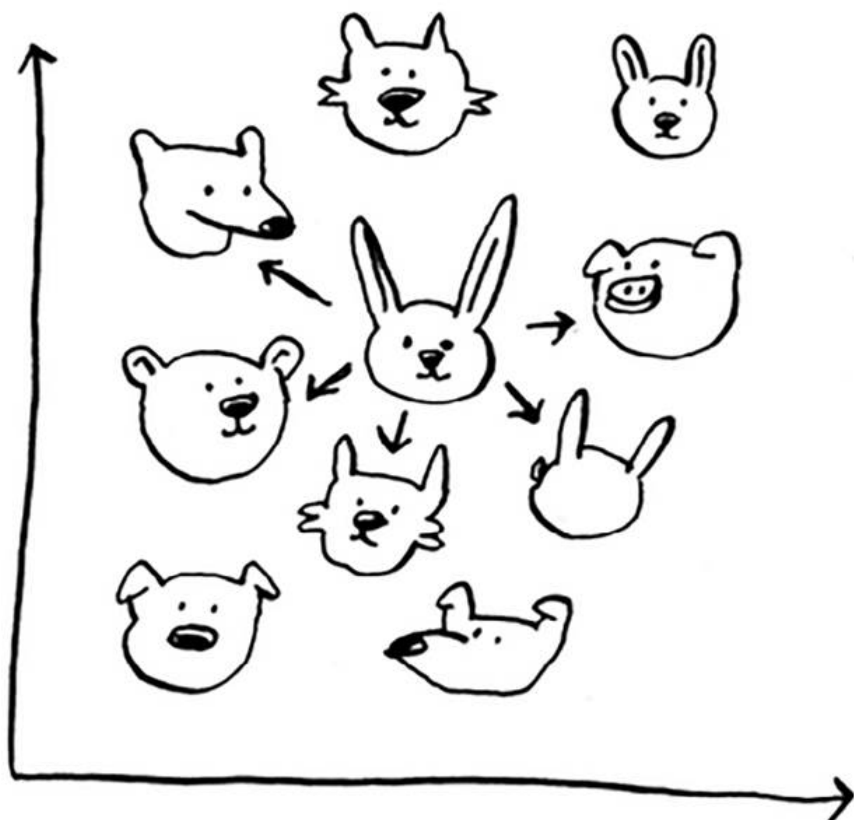
10.2 [illegible]

[illegible]Netflix[illegible]

[illegible]



[illegible]Priyanka[illegible]



□□□□□□□□□□□□□Justin□JC□Joey□Lance□Chris□□Priyanka□□□□□□□□□□□□□□□□Priyanka□□□□

□□□□□□□□□□□□□□□□□□□□□□□□□□Justin□□□□□□□□□□□□Priyanka□



□□□

10.2.1 □□□□

[illegible]

	A	B	C
个头	2	2	4
红的程度	2	1	5

[illegible]

[illegible]A[illegible]B[illegible]

$$\sqrt{(X_1 - X_2)^2 + (Y_1 - Y_2)^2}$$

[illegible]A[illegible]B[illegible]

$$\sqrt{(2-2)^2 + (2-1)^2}$$

$$= \sqrt{0 + 1}$$

$$= \sqrt{1}$$

$$= 1$$

A[illegible]B[illegible]1[illegible]

□□□□□□□□□□□□A□B□□□

□□□□□□□Netflix□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□
□□□□□□□□□□□□□□

(2, 2)

(?, ?)

[illegible]

[illegible]

	PRIYANKA	JUSTIN	MORPHEUS
喜剧片	3	4	2
动作片	4	3	5
生活片	4	5	1
恐怖片	1	1	3
爱情片	4	5	1

Priyanka [illegible] Justin [illegible] Morpheus [illegible] 2 [illegible]

□□□□□□□□□5□□□□□□

$$(2, 2)$$

$$(3, 4, 4, 1, 4)$$

□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□

$$\sqrt{(a_1 - a_2)^2 + (b_1 - b_2)^2 + (c_1 - c_2)^2 + (d_1 - d_2)^2 + (e_1 - e_2)^2}$$

□□□□□□5□□□□2□□□□

□□5□□□□□□□□□
□□□□□□□□□□□□□□□□□□□□□□□□

$$\sqrt{(3-4)^2 + (4-3)^2 + (4-5)^2 + (1-1)^2 + (4-5)^2}$$

$$= \sqrt{1 + 1 + 1 + 0 + 1}$$

$$= \sqrt{4}$$

$$= 2$$

□□Priyanka□Justin□□□□

Priyanka□Justin□□□Priyanka□Morpheus□□□□□□□□□□□□□□□□□□□□
□□□□□□□

Priyanka□Morpheus□□□□24□□□□□□□□□□□□□□Priyanka□□□□□□□
Justin□□□Morpheus□

□□□□□□□□Priyanka□□□□□□□□□□□□□Justin□□□□□□□□□□□□
Priyanka□□□□□□□□□□□□□□□□□□□□□□

□□□□Netflix□□□Netflix□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□
□□□□□□□□□□□□□□□□□□□□□□□□Netflix□□□□□□□□□□□□□□□□□□□

□□

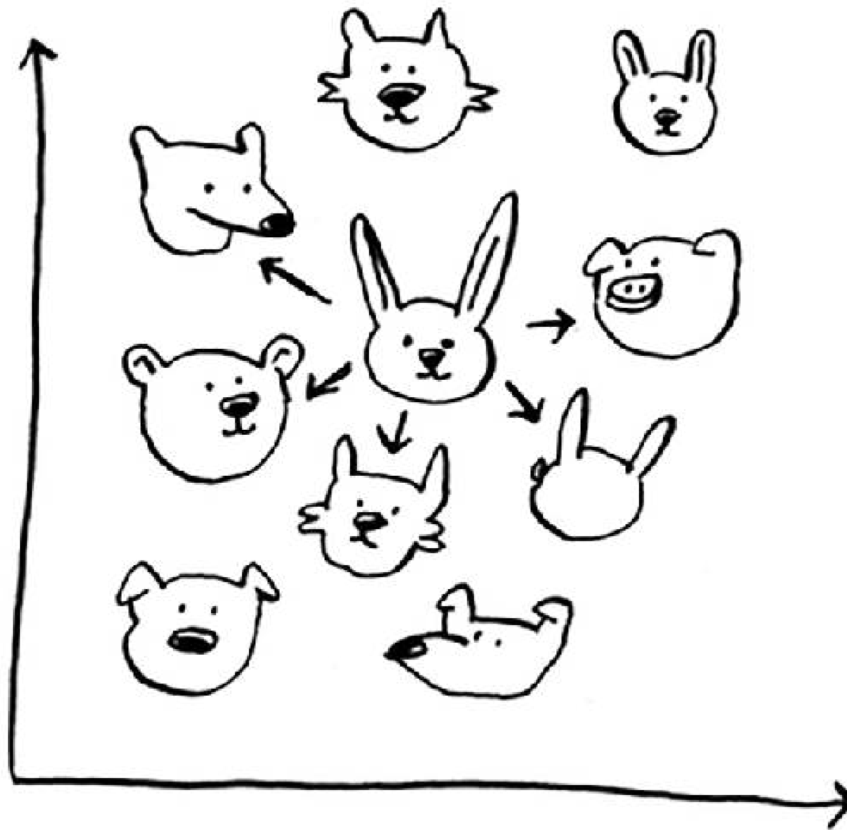
10.1 □Netflix□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□
□□□□□□□□□□□□——Yogi□Pinky□□□□□□□□□□□□□□□Yogi□□□□□□□□5
□□□Pinky□□□□□□□□□□□□□5□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□
□□□□□□□□□□□□□□

10.2 □□Netflix□□□□□□□□□□□□□Quentin Tarantino□Wes
Anderson□□Netflix□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□
□□□□□□□□□□□□□

10.2.2 □□

□□□□□□□Priyanka□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□5□□□

□□□□□□□□□□□□5□□□□□□□□□□□□5□□□□□□□□□□□2□□10□□10 000
□□□□□□□□□□□K□□□□□□5□□□□□□□



□□□□□□Priyanka□□□□□*Pitch Perfect* □□□□□Justin□JC□Joey□Lance□Chris□□□□□□□□□□□

JUSTIN : 5

JC : 4

JOEY : 4

LANCE : 5

CHRIS : 3

□□□□□□□□□□□□□□□□4.2□□□□□□ □regression□□□□□□KNN□□□□□□□□——□□□□□□

- □□□□□□□
- □□□□□□□□□□□□□□□□

□□□

- □□□□1□5□1□□□□□□□5□□□□□□□□□
- □□□□□□□□□□□□□□□□□1□□□0□□
- □□□□□□1□□□□0□□□□□□



□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□

A. (5, 1, Ø) = 3ØØ 条　　B. (3, 1, 1) = 225 条

C. (1, 1, Ø) = 75 条　　D. (4, Ø, 1) = 2ØØ 条

E. (4, Ø, Ø) = 15Ø 条　　F. (2, Ø, Ø) = 5Ø 条

[illegible]KNN[illegible]K[illegible]4[illegible]4[illegible]

$$(4, 1, \emptyset) = ?$$

[illegible]A[illegible]B[illegible]D[illegible]E[illegible]

A. 1 ←

B. 2 ←

C. 9

D. 2 ←

E. 1 ←

F. 5

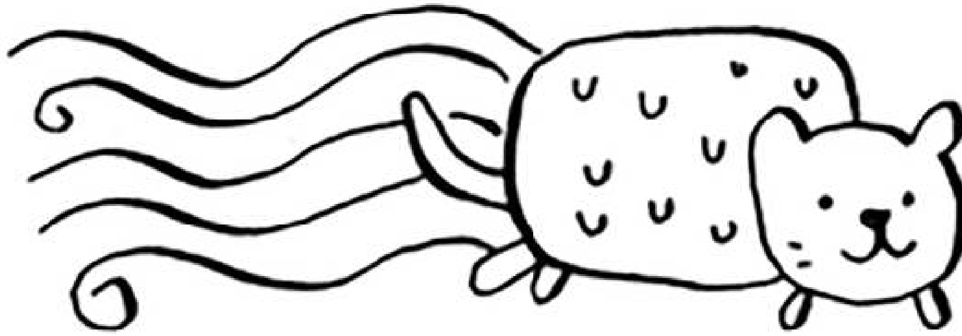
[illegible]218.75[illegible]

[illegible]

[illegible] [illegible]cosine similarity[illegible]Manmohan Desai[illegible]*Amar Akbar Anthony* [illegible]Paul[illegible]5[illegible]Rowan[illegible]4[illegible]

[illegible]KNN[illegible]

10.2.3 [illegible]



[illegible]

[illegible]2[illegible]3[illegible]KNN[illegible]

- [illegible]
- [illegible]

[illegible]*The Wire* [illegible]*House Hunters* [illegible]*House Hunters* [illegible]Netflix[illegible]

[illegible]

[illegible]

[illegible]

10.3 Netflix[illegible]5[illegible]

10.3 [illegible]

KNN[illegible]



10.3.1 OCR

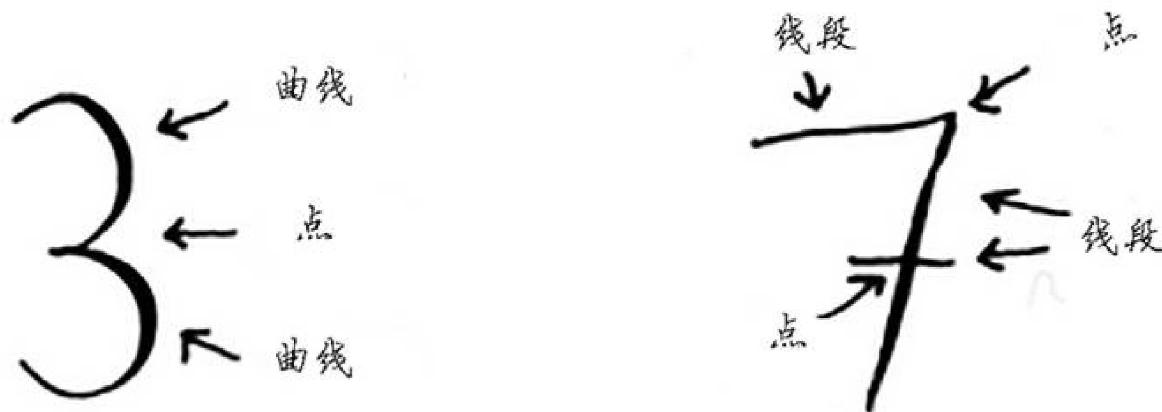
OCR□□□□□□□□□ □optical character recognition□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□Google□□OCR□□□□□□□□□OCR□□□□□□□□□□□□□□□□□□□□□□□□□

□□□□□□□□□□□□□□□□□□□KNN□

(1) [illegible]

(2) [illegible]

[illegible]OCR[illegible]



[illegible]

[illegible]OCR[illegible]KNN[illegible]Facebook[illegible]

OCR[illegible] [illegible]training[illegible]

10.3.2 [illegible]

[illegible]——[illegible] [illegible]Naive Bayes classifier[illegible]

主题	是不是垃圾邮件
"RESET YOUR PASSWORD"	不是
"YOU HAVE WON 1 MILLION DOLLARS"	是
"SEND ME YOUR PASSWORD"	是
"NIGERIAN PRINCE SENDS YOU 10 MILLION DOLLARS"	是
"HAPPY BIRTHDAY"	不是

□□□□□□□□□□□“collect your million dollars now!”□□□million□□KNN□□□

□□□



10.3.3 □□□□□□□

[illegible]

10.4 [illegible]

[illegible]KNN[illegible]

- KNN[illegible]
- [illegible]
- [illegible]
- [illegible]
- [illegible]KNN[illegible]

[illegible] 11 [illegible] [illegible]

[illegible]

- [illegible]10[illegible]
- [illegible]

11.1 [illegible]

[illegible]Facebook[illegible]Facebook[illegible]

[illegible] binary search tree [illegible]

[illegible]

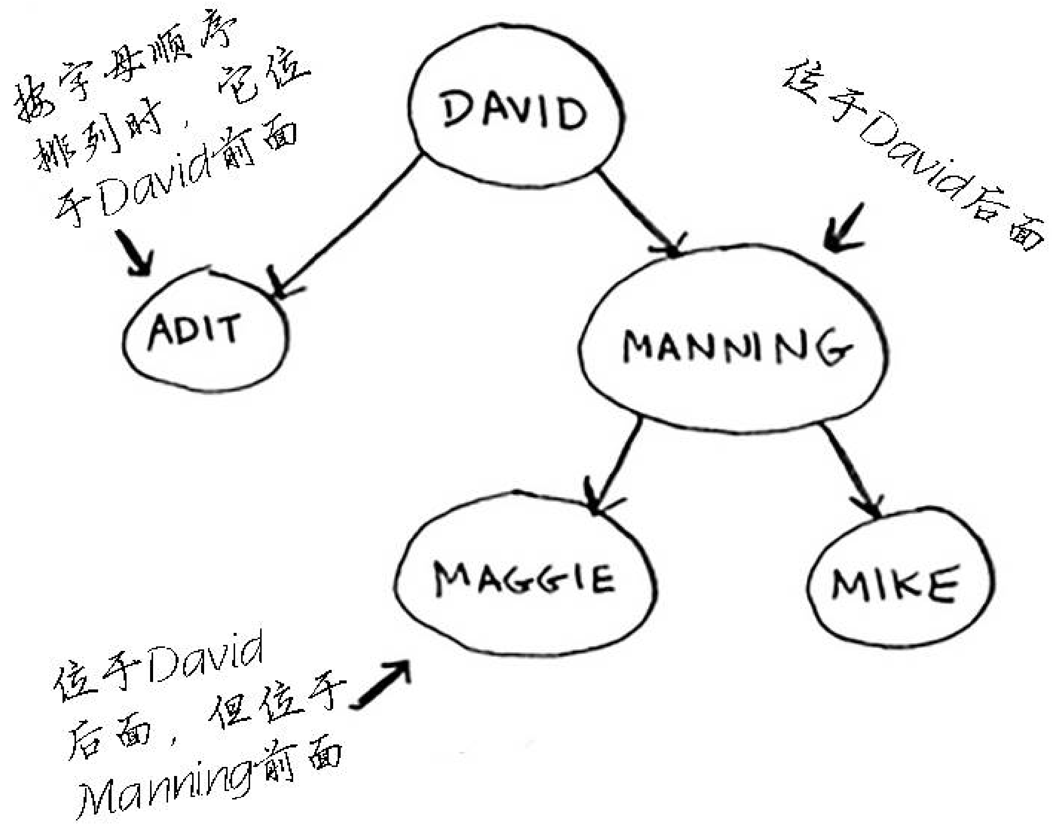
DAVID

ADIT

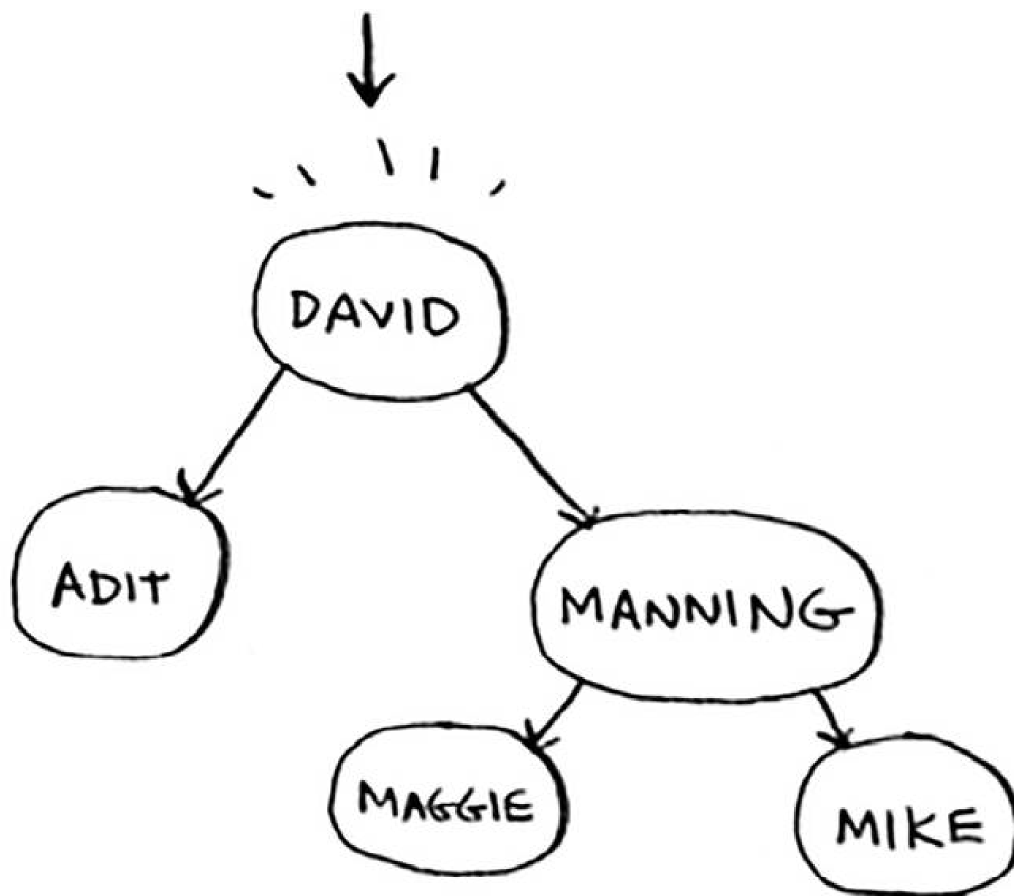
MANNING

MAGGIE

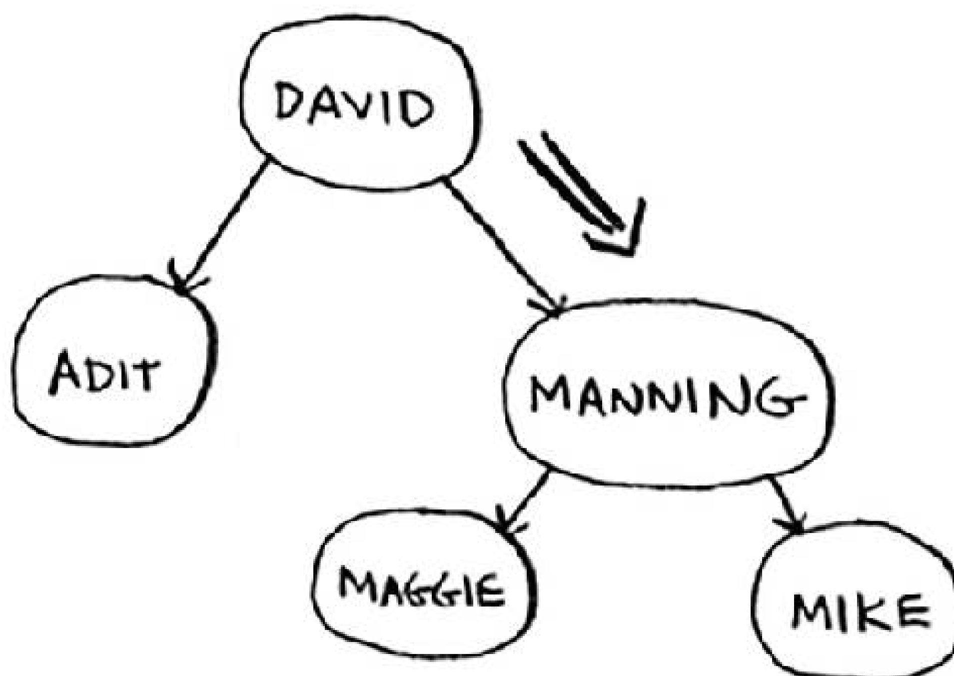
MIKE


按字母顺序排列时，它位于David前面
DAVID
位于David后面
ADIT
MANNING
MAGGIE
MIKE
位于David后面，但位于Manning前面

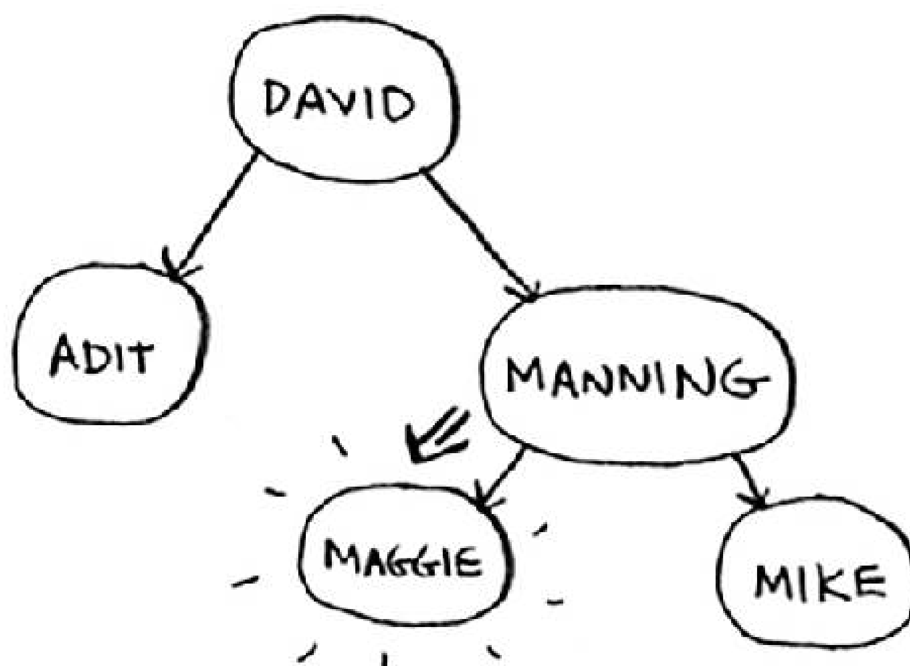
□□□□□□Maggie□□□□□□□□□□□□□



Maggie[illegible]David[illegible]



Maggie□□Manning□□□□□□□□□□□



□□□□□Maggie□□□□□□□□□□□□□□□□□□□□□□□□□□□□□$O(\log n)$□□□□□□□□□□□□□$O(n)$□□□□□□□□□□□□□□□□□□□□□□□□□□□$O(\log n)$□□□

	数组	二叉查找树
查找	$O(\log n)$	$O(\log n)$
插入	$O(n)$	$O(\log n)$
删除	$O(n)$	$O(\log n)$

□□□□□□□□□□□□□□□□□□□□□□□□□□□□□“□□□□□□□□”□□□□□□□□□□□□□□□□□□□□□□□□□$O(\log n)$□□□□□□□□□□□□□□□□□□□□□

B

B

11.2

HI
THERE

A.

HI
ADIT

B.

THERE
WE
GO

C.

□□□□□□□□□□□□□□□□□□

HI	A, B
THERE	A, C
ADIT	B
WE	C
GO	C

□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□hi□□□□□□□□□□□□□□□□□□□
□□□□□hi□

□□□□□□A□B□□hi□□□□□□□□□□□□□□□□□□□□□□there□□□□□A□C□□ □inverted index□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□

11.3 □□□□□

□□□□□□□□□□□□□□□□□□□□□□Better Explained□□[1] □□□□□□□□□□□□□□□□□□□□□□□□□

> [1] □□Kalid□□□Better Explained□□□□“An Interactive Guide to the Fourier Transform”□□□□http://mng.bx/874X □

□□MP3□□□□□□□

□□□□□□□□□□□□JPG□□□□□□□□□□□□□□□□□□□□□□□□□□□DNA□□□

□□□□□□□□□□Shazam□□□□□□□□□□□□□□□□□□□□□□□□□□□

11.4 □□□□

[illegible]

[illegible]$O(n \log n)$[illegible]$O(n)$[illegible]$O(n)$[illegible]

[illegible]

- [illegible] [illegible]1000[illegible]500[illegible]

- [illegible] [illegible]10[illegible]5[illegible]A[illegible]10[illegible]B[illegible]1[illegible]50[illegible]B[illegible]A[illegible]

[illegible]

11.5 MapReduce

[illegible] [illegible]MapReduce[illegible]Apache Hadoop[illegible]

11.5.1 [illegible]

[illegible]SQL[illegible]MySQL[illegible]Hadoop[illegible]MapReduce[illegible]

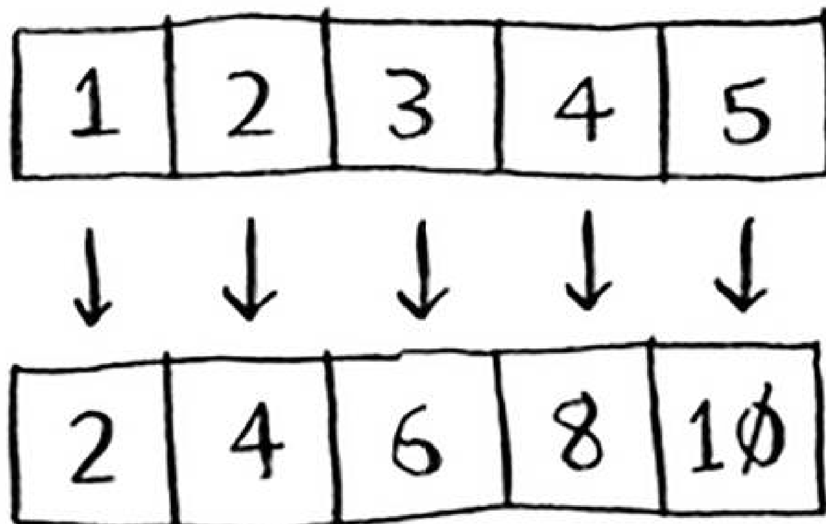
[illegible]100[illegible]10[illegible]100[illegible]

[illegible]MapReduce[illegible]map [illegible]reduce [illegible]

11.5.2 [illegible]

[illegible]

```
>>> arr1 = [1, 2, 3, 4, 5]
>>> arr2 = map(lambda x: 2 * x, arr1)
[2, 4, 6, 8, 10]
```



arr2 [illegible][2, 4, 6, 8, 10] [illegible]arr1 [illegible]

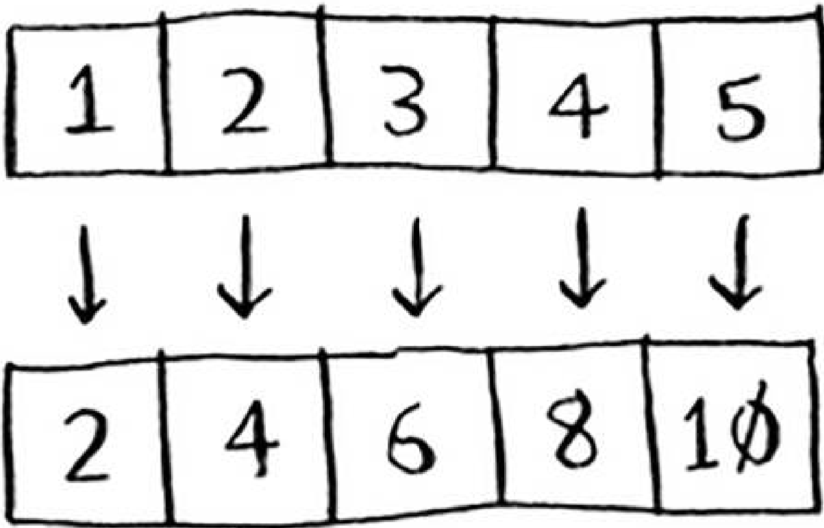
```
>>> arr1 = # A list of URLs
>>> arr2 = map(download_page, arr1)
```

[illegible]URL[illegible]URL[illegible]arr2 [illegible]URL[illegible]1000[illegible]URL[illegible]

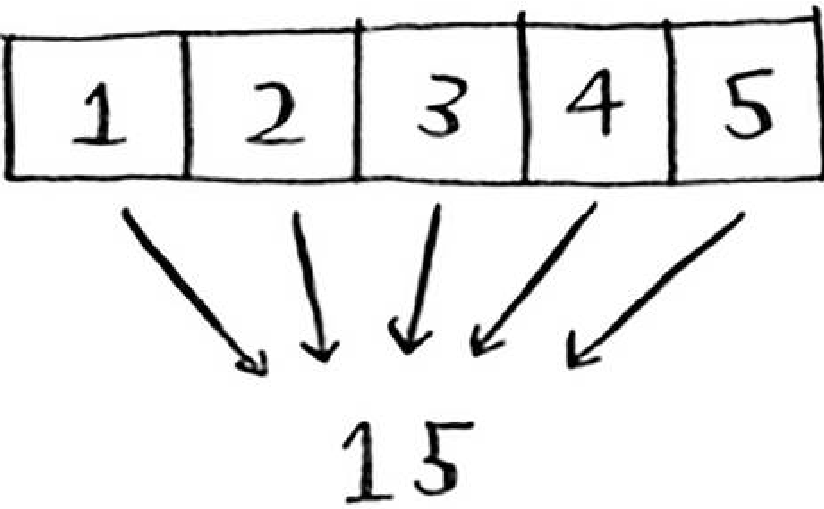
□□□100□□□□□□map □□□□□□□□□□□□□□□□□□□□□□□□□□□□□□100□□
□□□□□□□□□□□□□□MapReduce□“□□”□□□□□□□□

11.5.3 □□□□

□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□



□□□□□□□□□□□□□□□□



□□□□□□□□

```
>>> arr1 = [1, 2, 3, 4, 5]
>>> reduce(lambda x,y: x+y, arr1)
15
```

1 + 2 + 3 + 4 + 5 = 15

MapReduce
MapReduce

11.6 HyperLogLog

Reddit

Google

bit.ly
URL URL

[illegible]Google[illegible]
[illegible]

[illegible]adit.io[illegible]

`adit.io` [illegible]$O(1)$[illegible]

[illegible]Google[illegible]Reddit[illegible]
bit.ly[illegible]

11.6.1 [illegible]

[illegible]

- [illegible]Google[illegible]“[illegible]”[illegible]
- [illegible]“[illegible]”[illegible]

[illegible]Google[illegible]URL[illegible]
bit.ly[illegible]“[illegible]”

11.6.2 HyperLogLog

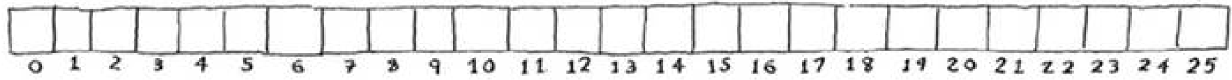
HyperLogLog[illegible]Google[illegible]Amazon[illegible]Google[illegible]Google[illegible]

HyperLogLog[illegible]

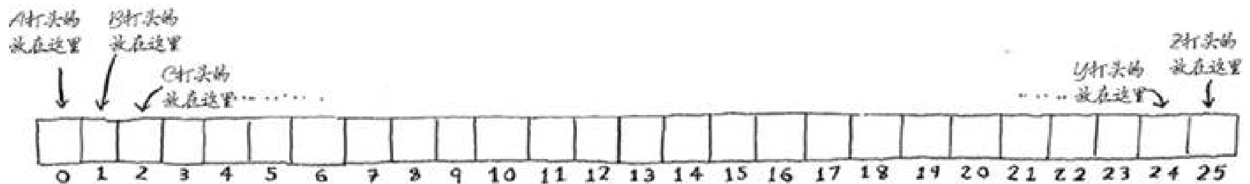
[illegible]

11.7 SHA[illegible]

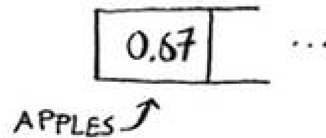
[illegible]5[illegible]



[illegible]



[illegible]



[illegible] $O(1)$ [illegible]

[illegible]

11.7.1 [illegible]

[illegible] secure hash algorithm [illegible] SHA [illegible] SHA [illegible]

“hello” ⇒ 2cf24db…

[illegible] SHA [illegible] —— [illegible] SHA [illegible]

[illegible] SHA [illegible]

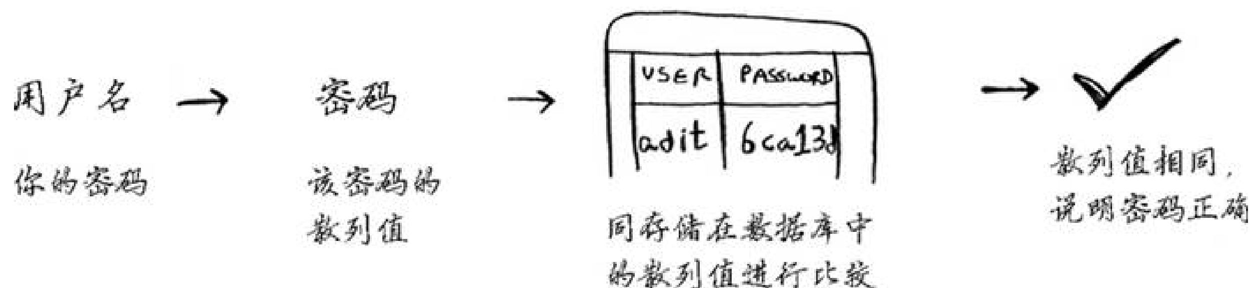
[illegible]

SHA [illegible]

[illegible]SHA[illegible]4 GB[illegible]SHA[illegible]

11.7.2 [illegible]

SHA[illegible]Gmail[illegible]Google[illegible]SHA[illegible]Google[illegible]



Google[illegible]SHA[illegible]

abc123 → 6ca13d

[illegible]

? ← 6ca13d

[illegible]Gmail[illegible]SHA[illegible]

SHA[illegible]SHA-0[illegible]SHA-1[illegible]SHA-2[illegible]SHA-3[illegible]SHA-0[illegible]SHA-1[illegible]SHA[illegible]SHA-2[illegible]SHA-3[illegible]bcrypt[illegible]

11.8 [illegible]

SHA[illegible]

dog → cd6357

[illegible]

dot → e392da

[illegible]

[illegible]Simhash[illegible]Simhash[illegible]

- Google[illegible]Simhash[illegible]
- [illegible]Simhash[illegible]
- Scribd[illegible]Simhash[illegible]·[illegible]

[illegible]Simhash[illegible]

11.9 Diffie-Hellman[illegible]

[illegible]Diffie-Hellman[illegible]

[illegible]a[illegible]1[illegible]b[illegible]2[illegible]“4,15,7”[illegible]“d,o,g”[illegible]——[illegible]

[illegible]“9,6,13,13,16 24,16,19,13,5”[illegible]a = 1[illegible]b = 2[illegible]

□□□□

□□□□□□□□□□□□□□□□□a = 2，b = 3□□

□□
□□Diffie-Hellman□□□□□□□□□□□□

- □□□□□□□□□□□□□□□□□□□□□□□□□□□□
- □□□□□□□□□□□□□□

Diffie-Hellman□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□
□□□
□□

Diffie-Hellman□□□□□□□RSA□□□□□□□□□□□□□□□□□□□□□□□Diffie-Hellman□□□□□□□□□□□□□□□□□□□

11.10 □□□□

□□□□□□□□□□□□□□□□□□□□□□□□□□□□□

□□
□□□□□□□□2□□□□□□□1□□□□5□□□□□□□□□□□3□□□□□□□2□□□□2□□
□□□□11□□□□20□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□

□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□

□□□
□□□1□□□□□□□□□□□□□□□□2□□□□□□□□□□□□□□□1.5□□□□□□1□□□□□□□
□□□□□□□□□□□□□□□□□500□300□□□□□□□50□□□□□□□□□1500□□□□
□□□□□□□□□□□□□□□□□□□

□□□□□□□□□□□□□□□□□□□□□□□□□□

□□□
□□□

□□□□□□Simplex□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□
□□□□

11.11 □□

□□□□□□□□10□□
□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□

□□□□

□ 1 □

1.1 7□□

1.2 8□□

1.3 $O(\log n)$□

1.4 $O(n)$[illegible]

1.5 $O(n)$[illegible]

1.6 $O(n)$[illegible]26[illegible]$O(n/26)$[illegible]O[illegible]O[illegible]$O(n+26)$[illegible]$O(n-26)$[illegible]$O(n*26)$[illegible]$O(n/26)$[illegible]$O(n)$[illegible]4.3[illegible]O[illegible]26[illegible]

[illegible] 2 [illegible]

2.1 [illegible]

2.2 [illegible]

2.3 [illegible]——[illegible]

2.4 [illegible]Adit B[illegible]Facebook[illegible]

2.5 [illegible]——[illegible]

Facebook[illegible]B[illegible]

[illegible] 3 [illegible]

3.1 [illegible]

- [illegible]greet [illegible]name [illegible]maggie [illegible]
- [illegible]greet [illegible]greet2 [illegible]name [illegible]maggie [illegible]
- [illegible]greet [illegible]
- [illegible]greet2 [illegible]
- [illegible]greet [illegible]

3.2 [illegible]

[illegible] 4 [illegible]

4.1

```
def sum(list):
  if list == []:
    return 0
  return list[0] + sum(list[1:])
```

4.2

```
def count(list):
  if list == []:
    return 0
  return 1 + count(list[1:])
```

4.3

```
def max(list):
  if len(list) == 2:
    return list[0] if list[0] > list[1] else list[1]
  sub_max = max(list[1:])
```

```
return list[0] if list[0] > sub_max else sub_max
```

4.4 □□□

□□□

4.5 $O(n)$□

4.6 $O(n)$□

4.7 $O(1)$□

4.8 $O(n^2)$□

□ 5 □

5.1 □□□

5.2 □□□□

5.3 □□□□

5.4 □□□

5.5 □□□□C□D□□□□□□□□

5.6 □□□□B□D□□□□□□□□

5.7 □□□□B□C□D□□□□□□□□

□ 6 □

6.1 □□□□□□□□2□

6.2 □□□□□□□□2□

6.3 A□□□□B□□□C□□□□

6.4 1——□□□2——□□□3——□□□4——□□□5——□□□□6——□□□□□7——□□□□

6.5 A□□□B□□□□C□□□C□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□

□ 7 □

7.1 A□8□B□60□C□□□□□□□□□□□□□□□□□□□□□□□□□□

□ 8 □

8.1 □□□

8.2 □□□

8.3 □□□

8.4 □□

8.5 □□

8.6 □□

8.7 □□

8.8 □□

□ 9 □

9.1 □□□□□□□□□□□□□MP3□□□□iPhone□□□□□□□□4500□□□

9.2 □□□□□□□□□□□□

9.3

	C	L	U	E	S
B	0	0	0	0	0
L	0	1	0	0	0
U	0	0	2	0	0
E	0	0	0	3	0

□ 10 □

10.1 □□□□□□ □normalization□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□Pinky□□□□□□□3□□Yogi□□□□□□3.5□□□□□□□□□□Pinky□□□□□□□□□□□□□3.5□□□□□□□□□□□□□□□□□□□□□□□□□

10.2 □□□□KNN□□□□□□□□□□□□□□□□□□3□□□——Joe□Dave□□□□□Wes Anderson□□□□Caddyshack□□□□□□3□□4□□5□□□□□□□□□□□□□□ (3 + 4 + 5) / 3 = 4□□□□Wes Anderson□□□□□□□□(3 + 4 + 5 + 5 + 5) / 5 = 4.4□□

10.3 □□N□□□□□□□sqrt(N)□□□□

□□□

□□□□□□□□□□□□□□□□□□□contact@turingbook.com□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□

□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□ebook@turingbook.com□

在这可以找到我们：

- 微博 @图灵教育 : 好书、活动每日播报
- 微博 @图灵社区 : 电子书和好文章的消息
- 微博 @图灵新知 : 图灵教育的科普小组
- 微信 图灵访谈 : ituring_interview，讲述码农精彩人生
- 微信 图灵教育 : turingbooks
